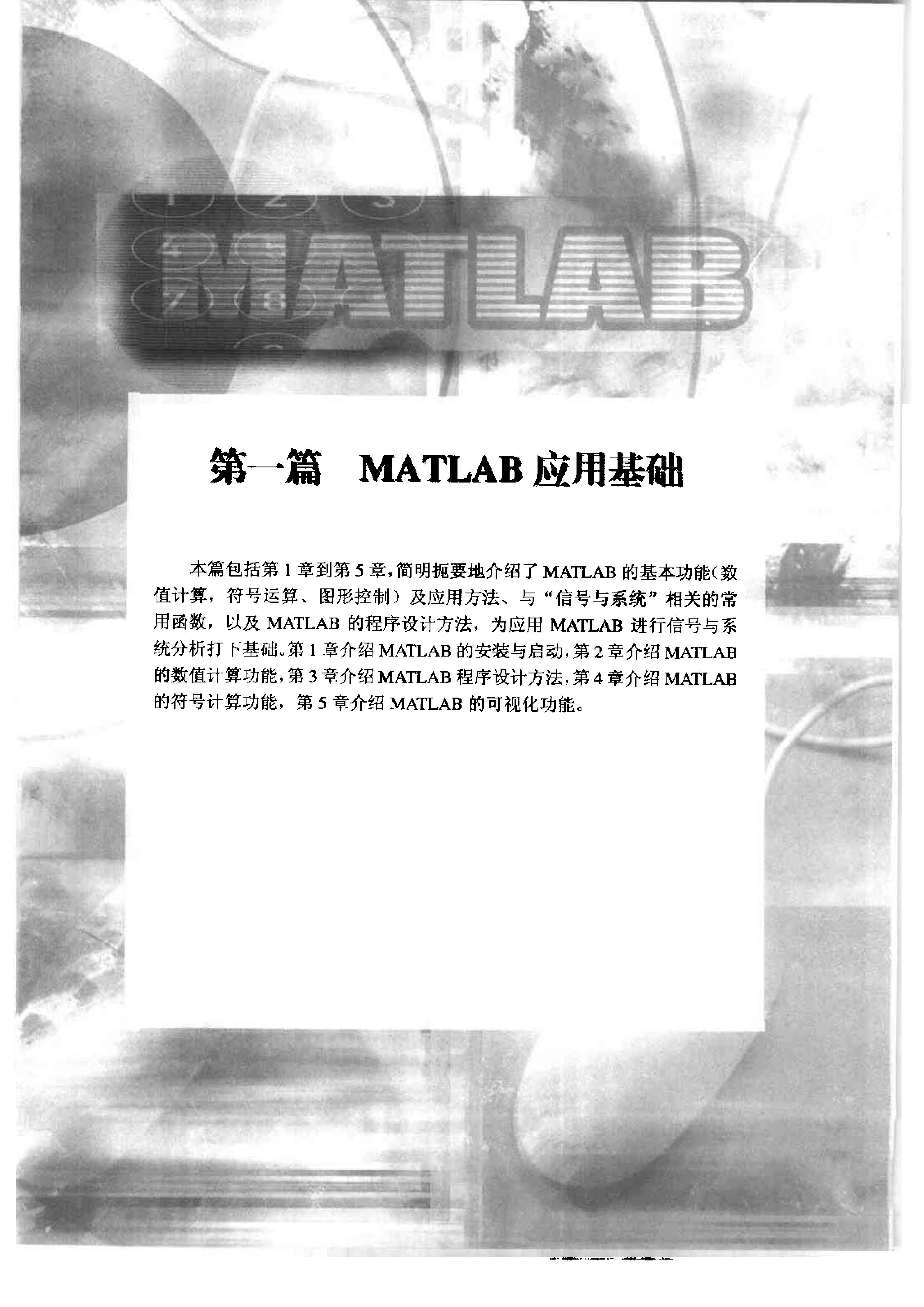
目 录

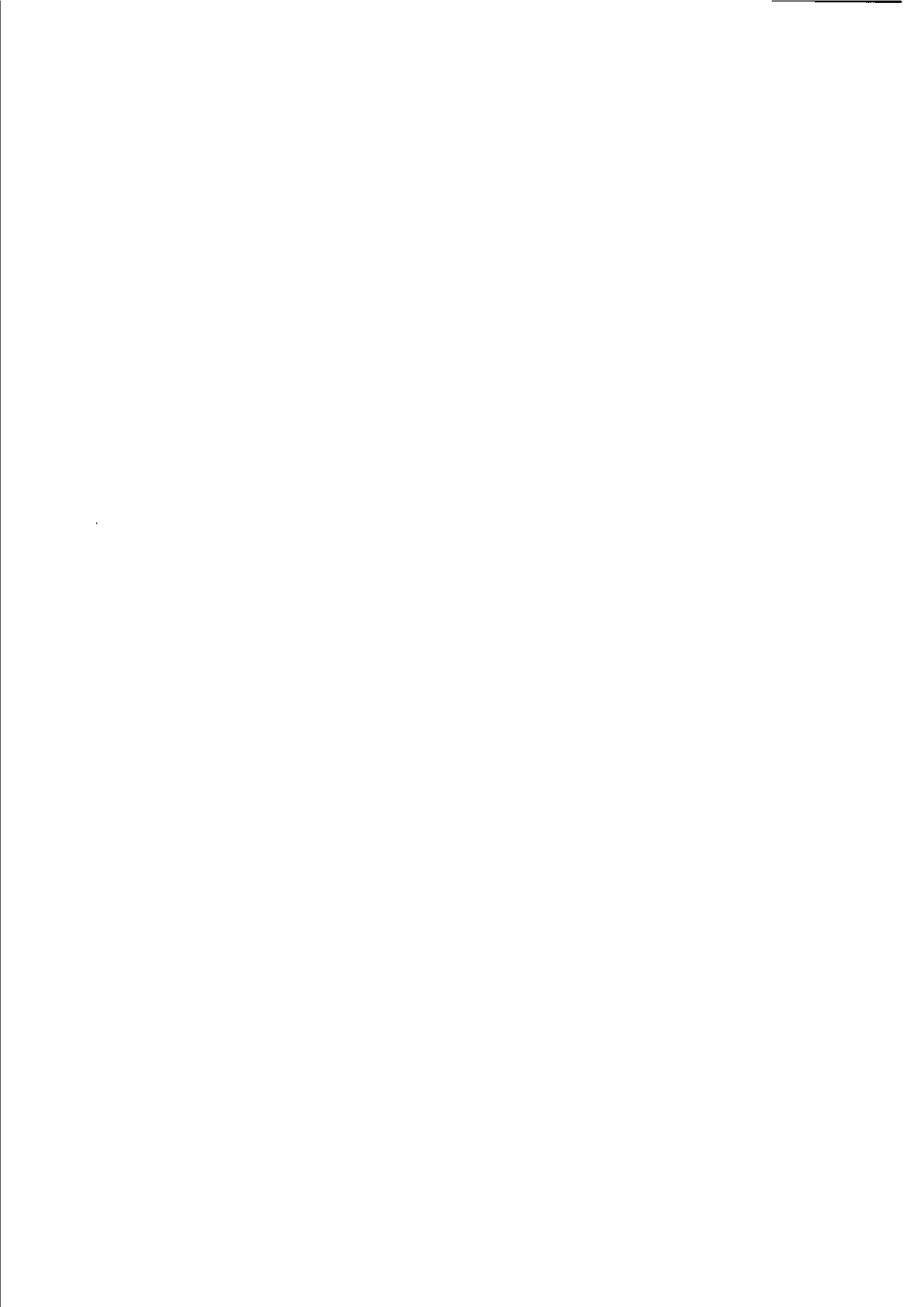
第一篇 MATLAB应用基础

第二篇 MATLAB 实现



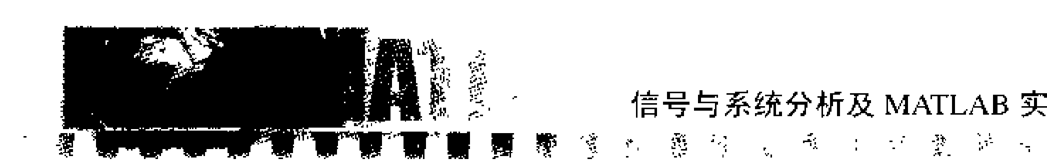
第一篇　MATLAB应用基础

本篇包括第1章到第5章,简明扼要地介绍了MATLAB的基本功能(数值计算，符号运算、图形控制)及应用方法、与“信号与系统”相关的常用函数，以及MATLAB的程序设计方法，为应用MATLAB进行信号与系统分析打下基础。第1章介绍MATLAB的安装与启动,第2章介绍MATLAB的数值计算功能,第3章介绍MATLAB程序设计方法,第4章介绍MATLAB的符号计算功能，第5章介绍MATLAB的可视化功能。



第 1 章　MATLAB 的安装、启动与退出





1.1 MATLAB 简介

在科学技术飞速发展的今天，计算机正扮演着越来越重要的角色。在进行科学研究与工程应用的过程中，科技人员往往会遇到大量繁重的数学运算和数值分析，传统的高级语言 BASIC、FORTRAN 及 C 语言等虽然能在一定程度上减轻计算量，但它们均要求应用人员具有较强的编程能力和对算法有深入的研究。

另外，在运用这些高级语言进行计算结果的可视化分析及图形处理方面，对非计算机专业的普通用户来说，仍存在着一定的难度。MATLAB 正是在这一应用要求背景下产生的数学类科技应用软件。它具有的顶尖的数值计算功能、强大的图形可视化功能及简洁易学的"科学便笺式"工作环境和编程语言，从根本上满足了科技人员对工程数学计算的要求，并将科技人员从繁重的数学运算中解放出来，因而，越来越受到广大科技工作者的普遍欢迎。

MATLAB 是 matrix 和 laboratory 前三个字母的缩写，意思是"矩阵实验室"，是 MathWorks 公司推出的数学类科技应用软件。其 DOS 版本（MATLAB 1.0）发行于 1984 年，现已推出了 Windows 版本（MATLAB 5.3）。经过十多年的不断发展与完善，MATLAB 已发展成为由 MATLAB 语言、MATLAB 工作环境、MATLAB 图形处理系统、MATLAB 数学函数库和 MATLAB 应用程序接口五大部分组成的集数值计算、图形处理、程序开发为一体的功能强大的系统。MATLAB 由"主包"和三十多个扩展功能和应用学科性的工具箱（Toolboxs）组成。

MATLAB 具有以下基本功能：

- 数值计算功能
- 符号计算功能
- 图形处理及可视化功能
- 可视化建模及动态仿真功能

MATLAB 语言是以矩阵计算为基础的程序设计语言，语法规则简单易学，用户不用花太多时间即可掌握其编程技巧。其指令格式与教科书中的数学表达式非常相近，用 MATLAB 编写程序尤如在便笺上列写公式和求解，因而被称为"便笺式"的编程语言。另外，MATLAB 还具有功能丰富和完备的数学函数库及工具箱，大量繁杂的数学运算和分析可通过调用 MATLAB 函数直接求解，大大提高了编程效率，其程序编译和执行速度远远超过了传统的 C 和 FORTRAN 语言，因而用 MATLAB 编写程序，往往可以达到事半功倍的效果。在图形处理方面，MATLAB 可以给数据以二维、三维乃至四维的直观表现，并在图形色彩、视角、品性等方面具有较强的渲染和控制能力，使科技人员对大量原始数据的分析变得轻松和得心应手。

正是由于 MATLAB 在数值计算及符号计算等方面的强大功能，使 MATLAB 一路领先，成为数学类科技应用软件中的佼佼者。目前，MATLAB 已成为国际上公认的最优秀的科技应用软件。MATLAB 的上述特点，使它深受工程技术人员及科技专家的欢迎，并很快成为应用学科计算机辅助分析、设计、仿真、教学等领域不可缺少的基础软件。目前，



在国外高等院校，MATLAB 已成为本科生、研究生必须掌握的基础软件，国内一些理工院校也已经把 MATLAB 作为学生必须掌握的一种软件，“教育部全国计算机专业课程指导委员会”已将 MATLAB 语言列为推荐课程。

1.2　MATLAB 的安装

MATLAB 既可在 PC 机单机环境下亦可在网络环境下安装运行，本书仅介绍 MATLAB 5.3 在 PC 机单机环境使用 Microsoft Windows 98 操作系统进行安装的情况。

MATLAB 5.3 对系统的基本要求为：

- Microsoft Windows 95/98/Me/NT/2000 操作系统
- 奔腾处理器
- 16 MB 以上内存（推荐更多内存）
- 1 GB 以上剩余硬盘空间
- 4 倍速以上光驱
- 8 位以上显卡
- 推荐使用声卡

在系统满足上述要求的情况下，即可进行 MATLAB 5.3 的安装，安装过程如下：

（1）将 MATLAB 5.3 光盘放入光驱，计算机将自动运行 MATLAB 安装程序，并显示如图 1-1 所示的安装对话框（MATLAB 的版权信息）。

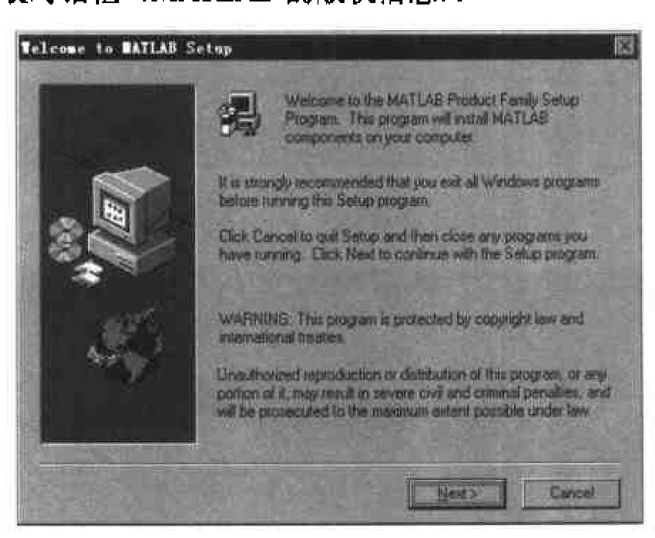


图 1-1　MATLAB 安装对话框一

（2）单击【Next】按钮继续下一步，屏幕显示如图 1-2 所示的安装对话框（MATLAB 的安装协议）。

（3）单击【No】按钮不接受协议，退出安装；单击【Yes】按钮接受协议，则屏幕显示如图 1-3 所示的安装对话框。

（4）在如图 1-3 所示的对话框中，分别输入用户名、公司名及 MATLAB 的产品序列号，然后单击【Next】按钮，则出现如图 1-4 所示的安装组件选择对话框。

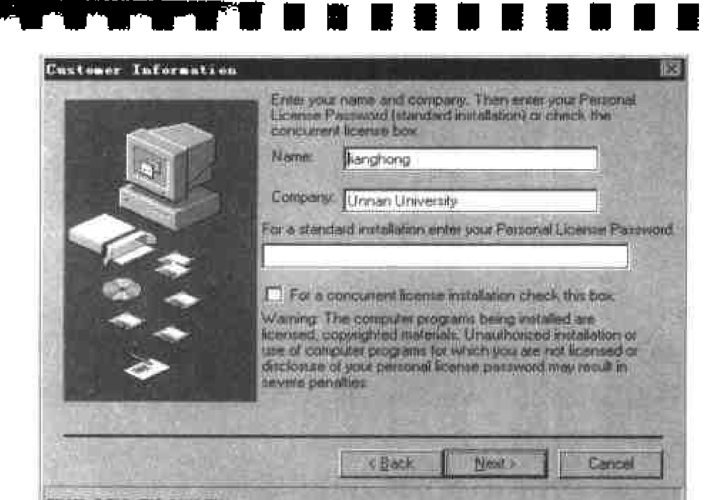


图 1-2 MATLAB 安装对话框二

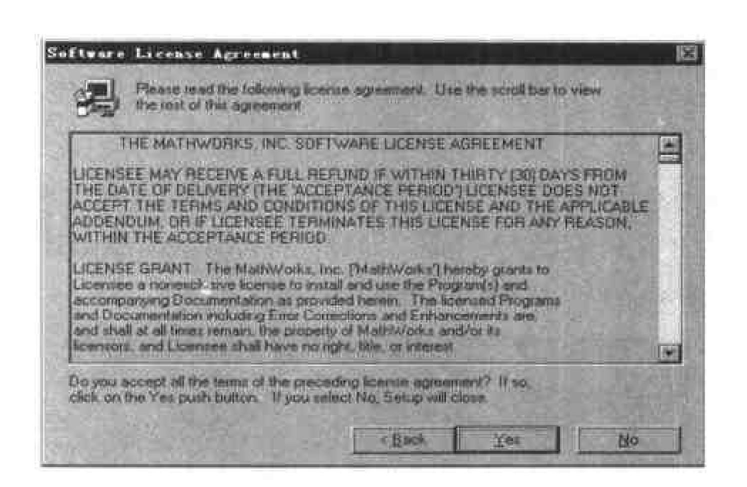


图 1-3 MATLAB 安装对话框三

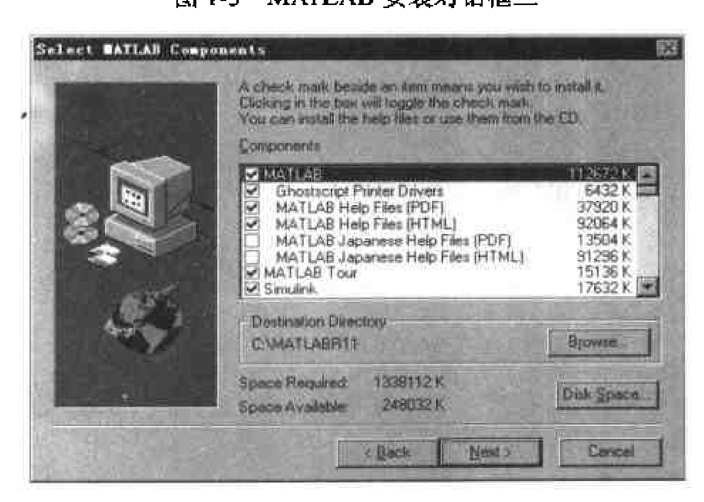


图 1-4 安装组件选择对话框

该对话框显示了 MATLAB 的所有可选安装组件（Components），其中包括“主包”

和三十多个工具箱（Toolbox），均通过该对话框中的复选框来选择。若读者安装 MATLAB 主要目的是进行信号处理与系统性能的分析，则推荐安装的组件如表 1-1 所示。通过单击安装组件选择对话框的【Browse】按钮用户还可以更改和设置安装目录。安装组件及安装目录选择设置完毕后，单击【Next】按钮，则屏幕显示如图 1-5 所示，系统正式开始安装 MATLAB 直至结束。MATLAB 安装完毕后，在系统桌面上将出现 MATLAB 的快捷方式及图标。

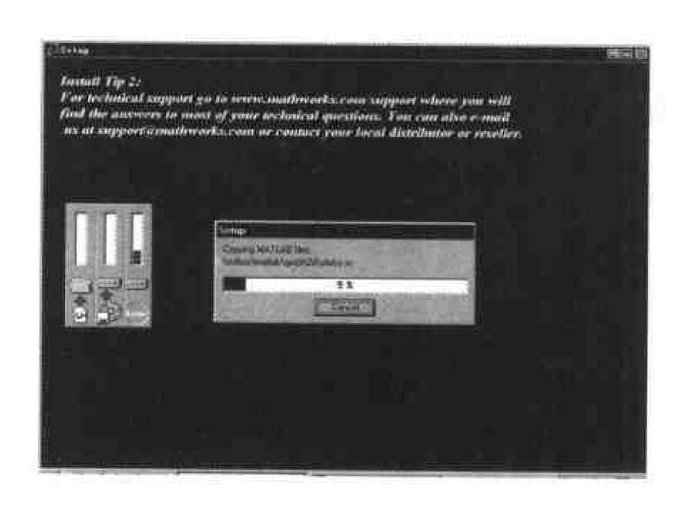


图 1-5　系统正在安装 MATLAB

表 1-1　信号与系统分析推荐安装组件

MATLAB	MATLAB 主包
MATLAB Help File [PDF]	Adobe 文档格式的帮助文件
MATLAB Help File [HTML]	超文本文档格式的帮助文件
Simulink	动态建模仿真软件包
Signal Processing Toolbox	信号处理工具箱
Image Processing Toolbox	图像处理工具箱
Control System Toolbox	控制工具箱
Wavelet Toolbox	小波工具箱
Communication Toolbox	通信工具箱
Extended Symbolic Math Toolbox	扩展符号数学工具箱

1.3　MATLAB 的启动与退出

1.3.1　MATLAB 的启动

MATLAB 的启动有如下两种方式：

方式一：双击 Windows 98 操作系统桌面上的 MATLAB 快捷方式，即可启动并打开

MATLAB 命令窗口。

方式二：单击【开始】菜单，依次指向【程序】→【MATLAB】→【MATLAB 5.3】，如图 1-6 所示，即可启动并打开 MATLAB 命令窗口。

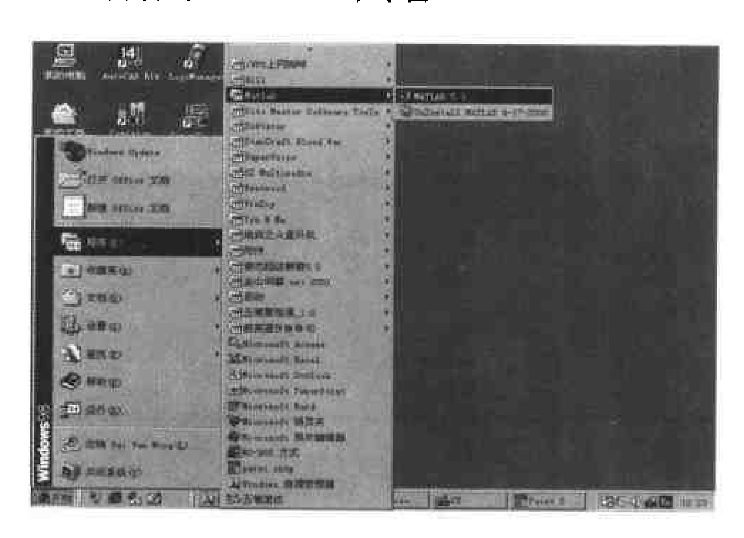


图 1-6　启动 MATLAB

1.3.2　命令窗口及操作

MATLAB 的命令窗口如图 1-7 所示，由标题栏、菜单栏、工具栏、工作区及状态栏组成。

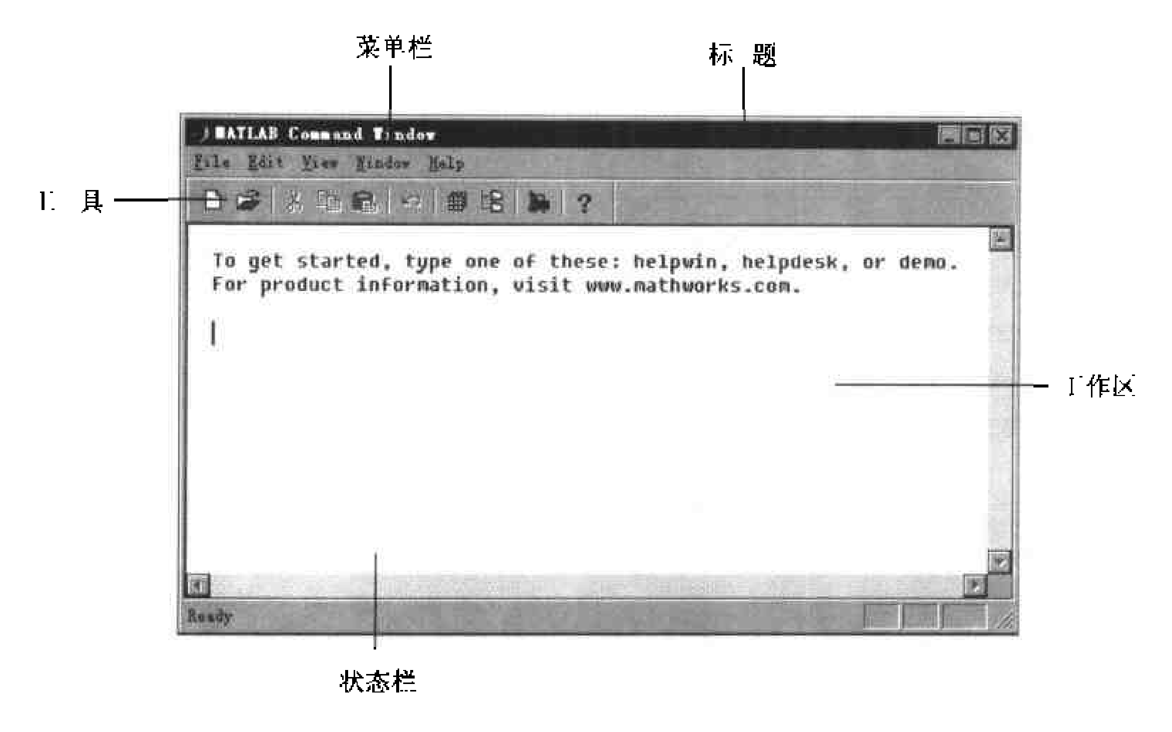


图 1-7　MATLAB 的命令窗口

菜单栏包括文件【File】、编辑【Edit】、浏览【View】、窗口【Window】和帮助【Help】五个菜单项。工具栏以图标方式为用户提供了 MATLAB 的常用命令及操作。工具栏图标及对应功能如图 1-8 所示。

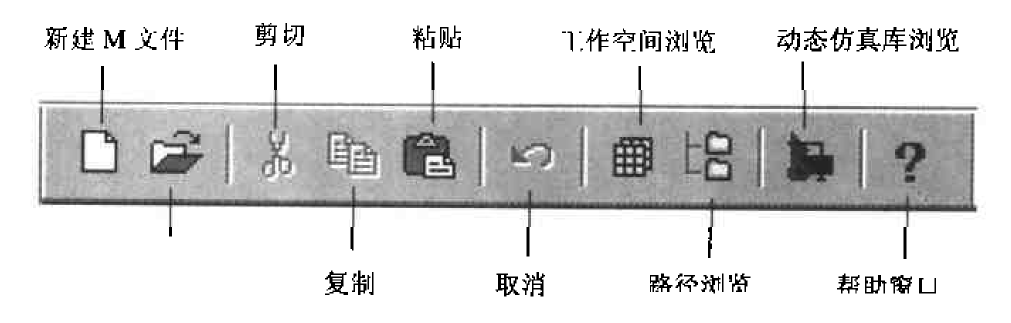


图 1-8 命令窗口工具栏

有关菜单栏和工具栏的详细使用，将在后述的运用过程中逐一介绍。

命令窗口的工作区是用户使用 MATLAB 的重要空间，MATLAB 在这里为用户提供了交互式的工作环境，即用户可随时输入命令，计算机即时给出运算结果。用户只需输入简单易学的 MATLAB 命令，即可进行诸如数值计算、符号运算和运算结果的可视化等复杂的分析和处理。但要注意，每一条命令或命令行键入后都要按【Enter】(回车）键，命令才会被执行。例如，在命令窗口的工作区直接输入如下字符：

```
a=ones(3,3)
```

然后按回车键，即可创建一个 3×3 且元素值为 1 的矩阵，并显示如下运行结果：

```
a =
      1     1     1
      1     1     1
      1     1     1
```

1.3.3 MATLAB 的退出

退出 MATLAB 非常简单，只需在 MATLAB 命令窗口内键入命令 quit 或单击命令窗口的【关闭】按钮即可。

1.4 如何获取帮助信息

MATLAB 为用户提供了强大的在线帮助功能，用户可以通过在线帮助轻松入门，并在帮助信息的指导下逐步熟练掌握 MATLAB 的应用。获取帮助信息有如下两种方式：

1. 利用帮助菜单获取帮助信息

- 单击 MATLAB 命令窗口菜单栏的【Help】菜单项，弹出帮助菜单选项，选择【Help Window】选项，则打开如图 1-9 所示的 MATLAB 帮助主题窗口。该窗口列出了 MATLAB 的所有帮助主题，双击相关主题即可打开有关该主题的进一步详细说明。

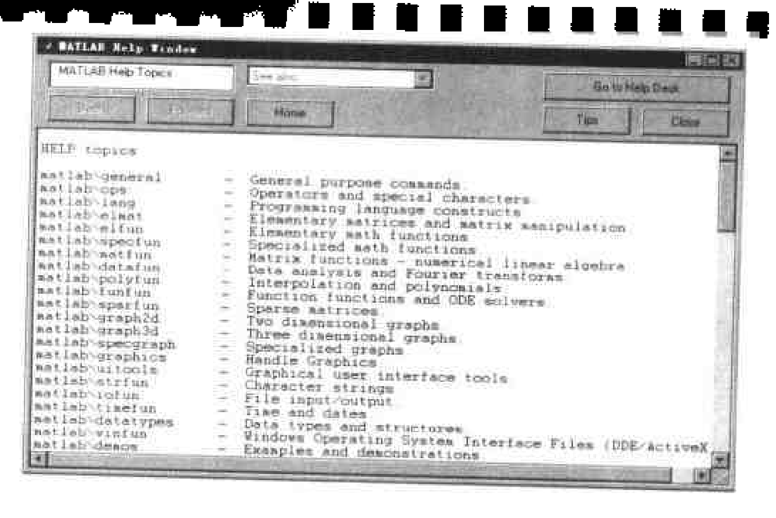


图 1-9　MATLAB 帮助主题窗口

- 单击 MATLAB 命令窗口菜单栏的【Help】菜单栏目，弹出帮助菜单选项，选择【Help Desk】选项，则打开图 1-10 所示的 MATLAB 帮助工作台。Helpdesk 以超文本方式为用户提供帮助信息，从基本的入门帮助到工具箱的使用。用户只需单击工作台中的相关主题，即可获取该主题的超文本格式的详细帮助信息。例如用户可以通过单击工作台的“Using MATLAB Graphics”主题来获取 MATLAB 有关图形处理的帮助信息。

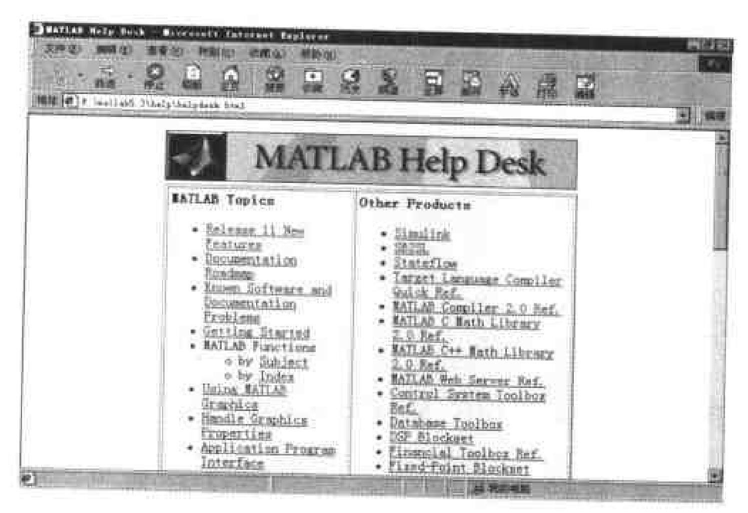


图 1-10　MATLAB 的帮助工作台

2. 通过命令窗口获取帮助信息

通过在命令窗口直接键入帮助命令也可获取 MATLAB 的在线帮助信息。帮助命令如下：

help	列出 MATLAB 的所有帮助主题
helpwin	打开 MATLAB 的帮助主题窗口
helpdesk	打开 MATLAB 的帮助工作台
help help	打开有关如何使用帮助信息的帮助窗口



help+函数名（或主题名）　　查询函数（或主题）的相关信息

例如，在 MATLAB 命令窗口中键入命令：

help ones

按下回车键，则命令窗口将显示关于 ones 函数的所有帮助信息，如图 1-11 所示。

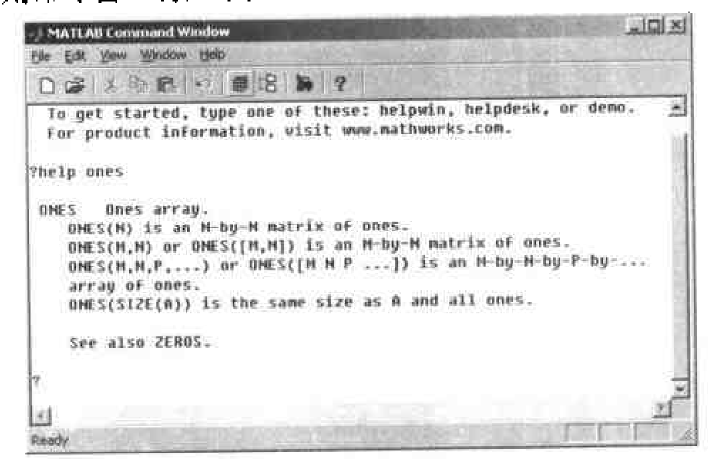


图 1-11　MATLAB 的帮助信息

1.5　使用演示功能（Demo）

MATLAB 带有生动直观的演示程序，以帮助用户学习和理解 MATLAB 的应用和强大功能。启动 MATLAB 的演示程序可通过如下两种方式：

方式一：选择【Help】菜单的【Examples and Demos】选项，即可打开 MATLAB 演示窗口，如图 1-12 所示。

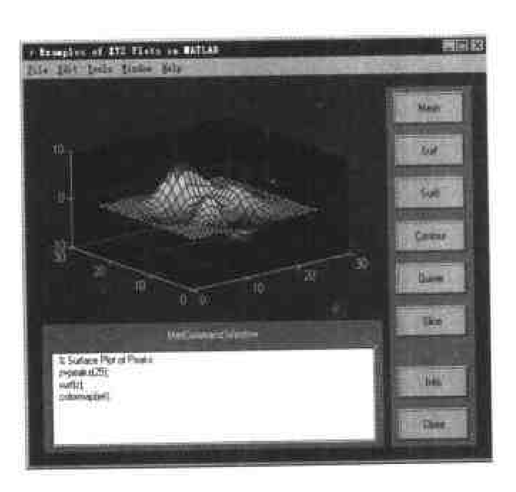


图 1-12　MATLAB 的演示窗口

方式二：在 MATLAB 的命令窗口中键入命令 Demo，也可打开图 1-12 所示的演示程序窗口。

该窗口左边列出了演示程序的主题，右边则列出了该主题的相关演示了项目。例如，

在窗口左边选择 Visualization（可视化）演示主题，然后单击窗口右边的“3-D Plots”演示项目，则打开如图 1-13 所示的演示窗口，运行 MATLAB 的三维图形处理演示程序。在形象直观的交互式演示过程中，用户不仅能更加深刻地体会到 MATLAB 强大功能，还能了解到演示程序的相应 MATLAB 命令。

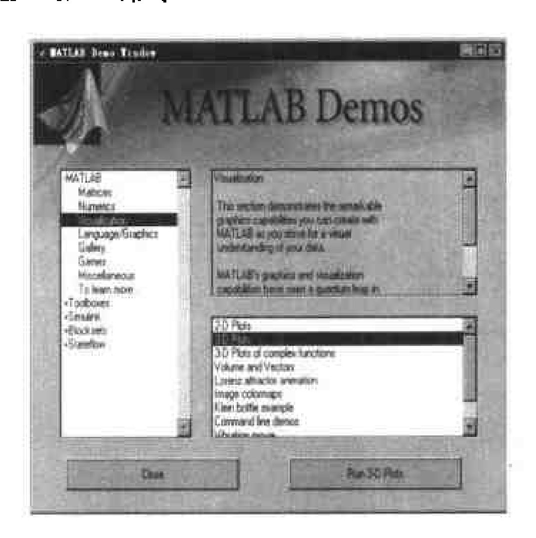


图 1-13 MATLAB 演示窗口

第 2 章　MATLAB 的数值计算功能



2.1 基本概念

1．变量

和其他高级语言一样，MATLAB 也是使用变量来保存信息。变量由变量名表示，变量的命名应遵循如下规则：

- 变量名必须以字母开头。
- 变量名可以由字母、数字和下划线混合组成。
- 变量名区分字母大小写。
- 变量名的字符长度不应超过 31 个。

在 MATLAB 中还存在着一些系统默认的固定变量，如表 2-1 所示，即在 MATLAB 语句中若出现固定变量名，则系统就将其赋予默认值。

表 2-1 MATLAB 的固定变量

变量名	默认值
i	虚数单位 $\sqrt{-1}$
j	虚数单位 $\sqrt{-1}$
pi	圆周率 π
inf	无穷大

MATLAB 的变量分为字符变量和数值变量两种，字符变量必须用单引号括起来。例如，用户可输入：

a='happy new year'

则表示将字符串'happy new year'赋值给字符变量 *a*。

若用户输入：

b=365

则表示将数值 365 赋值给数值变量 *b*。

和其他高级语言不同的是，MATLAB 使用变量时不需要预先对变量类型进行说明，MATLAB 会自动根据所输入的数据来决定变量的数据类型和分配存储空间。

2．数值

在 MATLAB 内部，每一个数据元素都是用双精度来表示和存储的，大约有 16 位有效数字。其数值有效范围约为 10^{-308}~10^{+308}。

但在进行数据输入输出时，MATLAB 却可以用不同的格式。如果参加运算的每一个元素均为整数，则 MATLAB 将用不加小数点的纯整数格式显示运算结果，否则，按默认的输出格式显示结果。MATLAB 的默认格式为 short 格式，该格式显示运算结果为保留小数点后 4 位有效数字。用户可以通过 format 命令改变输出格式为 long，以得到更多的有效数字(小数点后 14 位)。需要注意的是，数据输出格式的改变，并不影响该数据在 MATLAB 内部的存储精度。设置为 short 和 long 输出格式的命令分别为：

format short

format long

MATLAB 通常用十进制数来表示一个数，亦可用科学计数法来表示一个数，另外，MATLAB 还可以进行复数运算，复数可以由如下语句来产生：

c=a+i*b	（或 c=a+j*b）	将实部为 a 虚部为 b 的复数赋值给复变量 c
c=a*exp(i*b)	（或 c=a*exp(j*b)）	将模为 a 辐角为 b 的复数赋值给复变量 c

其中 i 、j 是虚数单位 $\sqrt{-1}$。

以下是 MATLAB 的各种合法的输入数据示例：

365	-18	0.000076	8.2467e-10
2.67i	1.02E3i	-7.6	-3.4502e+20

3．矩阵

矩阵是 MATLAB 进行数据处理和运算的基本元素，MATLAB 的大部分运算或命令都是在矩阵运算的意义下执行的。我们通常意义上的数量（标量）在 MATLAB 系统中是作为 1×1 的矩阵来处理的，而仅有一行或一列的矩阵在 MATLAB 中称为向量。

4．数组

在 MATLAB 中，数组也是一个非常重要的概念，矩阵在某些情况下可视为二阶的数值型数组。但是在 MATLAB 中，数组和矩阵运算规则却有着较大的区别。例如，两矩阵相乘和两数组相乘所遵循的运算规则就是完全不相同的。

5．函数

MATLAB 为用户提供了丰富且功能各异的函数，用户可以直接调用这些函数来进行数据处理。函数由函数名和参数组成，函数调用的格式为：

函数名（参数）

例如，若在 MATLAB 的命令窗口输入命令：

$a=\sin(b)$

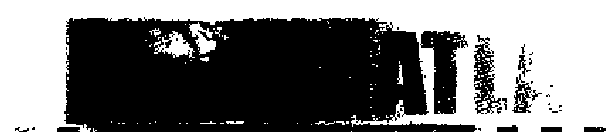
则表示计算 b 的正弦值并将其赋值给变量 a。

6．运算符

MATLAB 的基本运算为算术运算、关系运算、逻辑运算和特殊运算等，每一类运算都有自己专用的运算符，表 2-2、表 2-3、表 2-4 及表 2-5 分别列出了 MATLAB 的算术运算符、关系运算符、逻辑运算符、特殊运算符和其对应功能与示例。表中 ***A***、***B***、***X*** 代表矩阵。

表 2-2　MATLAB 的算术运算符

运算符	名称	指令示例	说明
+	加	***A+B***	若 *A*、*B* 为同维矩阵，则表示 ***A*** 与 ***B*** 对应元素相加；若其中一个矩阵为标量，则表示另一矩阵的所有元素加上该标量
-	减	***A-B***	若 *A*、*B* 为同维矩阵，则表示 ***A*** 与 ***B*** 对应元素相减；若其中一个矩阵为标量，则表示另一矩阵的所有元素减去该标量



（续表）

运 算 符	名 称	指令示例	说 明
*	矩阵乘	A*B	矩阵 A 与 B 相乘，A 和 B 均可是向量或标量，但 A 和 B 的维数必须符合矩阵乘法的定义
.*	数组乘	A.*B	矩阵 A 与 B 对应元素相乘，A 和 B 必须为同维矩阵或其中之一为标量
^	矩阵乘方	A^B	A、B 均为标量时，表示 A 的 B 次方幂；A 为方阵，B 为正整数时，表示矩阵 A 的 B 次乘积；A 为方阵，B 为负整数时，表示矩阵 A 逆矩阵的 B 次乘积；当 A、B 均为矩阵时，无定义
.^	数组乘方	A.^B	矩阵 A 的各元素与矩阵 B 的对应元素的乘方运算，即[A(i,j)^B(i,j)]，A、B 必须为同维矩阵
\	矩阵左除	A\B	方程 A*X=B 的解 X
.\	数组左除	A.\B	矩阵 B 的元素除以矩阵 A 的对应元素，A、B 必须为同维矩阵或其中之一为标量
/	矩阵右除	A/B	方程 X*A=B 的解 X
./	数组右除	A./B	矩阵 A 的元素除以矩阵 B 的对应元素，A、B 必须为同维矩阵或其中之一为标量
'	共轭转置	A'	矩阵 A 的共轭转置

表 2-3 MATLAB 的关系运算符

运 算 符	名 称
==	等于
~=	不等于
>	大于
<	小于
>=	大于等于
<=	小于等于

表 2-4 MATLAB 的逻辑运算符

运 算 符	名 称
&	逻辑和
|	逻辑
~	逻辑和

表 2-5 MATLAB 的特殊运算符

运 算 符	名 称	说 明
:	冒号	用于产生向量
[ ]	方括号	用于创建和表示矩阵
;	分号	或不显示中间结果
%	百分号	用于注释语句

（续表）

运　算　符	名　　称	说　　明
,	逗号	用于分隔矩阵列
=	等号	用于赋值
( )	圆括号	用于函数调用和指定运算顺序

有关各个运算符的使用，将在后述的应用中详细介绍。

7. MATLAB 的语句

MATLAB 采用命令行式的表达式语言，每一个命令行就是一条语句，其格式与书写的数学表达式十分相近，非常容易掌握。用户在命令窗口输入语句并按下回车键后，该语句就由 MATLAB 系统解释运行，并即时给出运行结果。MATLAB 的语句采用以下两种形式之一：

（1）表达式

（2）变量=表达式

表达式由变量名、常数、函数和运算符构成。例如下面表达式都是合法的表达式。

4*sin(2*t)　　sqrt(2)*exp(-i*4)　　s*a+b/c

在上述 MATLAB 语句的第一种形式中，表达式执行运算后产生的矩阵，将自动赋给名为“*ans*”的默认变量，并即时在屏幕上显示出来。变量“*ans*”的值将在下一次运行第一种形式的语句时被刷新。例如，输入下列语句

4*2+5/6

则表达式执行结果为：

ans =

8.8333

在 MATLAB 语句的第二种形式中，语句执行的结果是将表达式计算产生的矩阵，赋值给等号左边的变量，并存入内存。例如，若用户输入以下语句：

a=1+2+3+4+5

则执行结果为：

a =

15

注意

MATLAB 语句结尾若加上分号“;”，其作用是将计算结果存入内存，但不显示在屏幕上，反之，语句结尾若不加“;”，则表示在语句执行后，在将计算结果存入内存的同时，还将运算结果显示出来，读者可自行测试一下，语句

k=198/6;

和

k=198/6

运行结果有何不同。

2.2 矩阵的创建与保存

在 MATLAB 中，把由下标表示次序的标量数的集合称为矩阵，或称为数组。从数的集合的角度来看，数组和矩阵没有什么不同，但从运算角度看，矩阵运算和数组运算却遵循不同的运算规则。

矩阵是 MATLAB 进行数据处理的基本单元，MATLAB 的大部分运算都是在矩阵的意义上进行的，矩阵运算也是 MATLAB 最重要的运算。因此，对于初学者来说，掌握矩阵的生成、保存及基本运算是非常重要的。

在 MATLAB 中，矩阵可通过下列三种方法之一创建：

- 直接输入法；
- 利用 MATLAB 内部函数创建；
- 从外部数据文件（*.mat）装载并创建矩阵。

2.2.1 直接输入法创建矩阵

对于简单且维数较小的矩阵，创建矩阵的最佳方法就是从键盘直接输入矩阵，即按矩阵行的顺序输入矩阵各元素。在输入过程中必须遵循以下规则：

- 矩阵的所有元素必须放在方括号“[ ]”内；
- 矩阵元素之间必须用逗号“,”或空格隔开；
- 矩阵行与行之间用分号“;”或回车符隔开；
- 矩阵元素可以是任何不含未定义变量的表达式。

例如,用户在命令窗口内直接输入：

```
a=[1,2,3;4,5,6;7,8,9]
```

或

```
a=[1 2 3 ;4 5 6;7 8 9]
```

上述语句执行后，将创建相同的 3×3 的矩阵，并赋值给变量 a，运行结果显示为：

```
a =
      1     2     3
      4     5     6
      7     8     9
```

对于每行元素较多的矩阵，则可按矩阵书写的惯用格式输入。例如，在命令窗口内直接键入：

```
a=[1 1 1 1 1 1 1 1 1 1 1 1
   2 2 2 2 2 2 2 2 2 2 2 2
   3 3 3 3 3 3 3 3 3 3 3 3
   4 4 4 4 4 4 4 4 4 4 4 4
   5 5 5 5 5 5 5 5 5 5 5 5
   6 6 6 6 6 6 6 6 6 6 6 6]
```





键入回车执行后，将创建一个 6×12 的矩阵，并赋值给变量 *a*，运行结果显示如下：

```
a =
     1     1     1     1     1     1     1     1     1     1     1     1
     2     2     2     2     2     2     2     2     2     2     2     2
     3     3     3     3     3     3     3     3     3     3     3     3
     4     4     4     4     4     4     4     4     4     4     4     4
     5     5     5     5     5     5     5     5     5     5     5     5
     6     6     6     6     6     6     6     6     6     6     6     6
```

2.2.2　利用 MATLAB 函数创建矩阵

MATLAB 为用户提供了创建基本矩阵的函数，它们是：

- ones()函数
- zeros()函数
- rand()函数
- randn()函数
- eye 函数

ones()函数用于产生全为 1 的矩阵，ones(*n*)产生（*n*×*n*）维的全 1 阵，ones(*n*,*m*)产生 *n* 行 *m* 列的全 1 阵。例如，输入语句：

```
x=ones(3,4)
```

则运行结果为：

```
x =
     1     1     1     1
     1     1     1     1
     1     1     1     1
```

zeros()函数用于产生全为 0 的矩阵，zeros(*n*)产生（*n*×*n*）维的全 0 阵，zeros(*n*,*m*)产生 *n* 行 *m* 列的全 0 阵。例如，输入语句：

```
x=zeros(3,4)
```

则运行结果为：

```
x =
     0     0     0     0
     0     0     0     0
     0     0     0     0
```

函数 rand()用于产生在[0，1]区间均匀分布的随机阵，rand(*n*)产生（*n*×*n*）维的随机阵，rand(*n*,*m*)产生 *n* 行 *m* 列的随机阵。例如，输入语句：

```
a=rand(3,4)
```

则运行结果为：

```
a =
    0.9501    0.4860    0.4565    0.4447
```

```
0.2311    0.8913    0.0185    0.6154
0.6068    0.7621    0.8214    0.7919
```

函数randn()用于产生正态分布的矩阵，randn(*n*)产生（*n*×*n*）维的正态阵，randn(*n*,*m*)产生*n*行*m*列的正态阵。

函数eye()用于产生单位阵，eye (*n*)产生（*n*×*n*）维的单位阵。例如输入语句：

```
eye(3)
```

则运行结果为：

```
ans =
     1     0     0
     0     1     0
     0     0     1
```

2.2.3 利用外部数据文件（*.mat）保存和装载矩阵

在MATLAB的运用过程中，有时需要将矩阵数据长期保留下来，以备以后使用，这时就可以使用MAT文件来对矩阵数据进行保存，在需要时又将其装载到MATLAB环境中。

MAT文件（即扩展名为mat的文件）是MATLAB保存数据的一种标准格式的二进制文件。MAT文件的生成和调用由专用命令save和load来进行。

用户可以将已定义过的矩阵（变量）以MAT文件的格式存入到磁盘上，命令格式为：

```
save  路径\文件名 变量名
```

save命令可同时将多个矩阵（变量）保存到一个MAT文件中，此时变量名之间需用空格分开。

例如，用户输入命令：

```
save   c:\my a b
```

则表示将变量*a*、*b*以文件名my.mat保存到C盘根目录下。若路径默认，则MATLAB自动将变量保存到其默认的目录（work）中。

以MAT文件保存的矩阵，用户在使用MATLAB的任何时候均可用load命令装载到MATLAB的工作空中。例如，要将上述保存的变量*a*、*b*重新装载到MATLAB的工作空间，只需键入命令：

```
load c:\my
```

2.3 向量的生成

在MATLAB系统中，仅有一行或一列的矩阵称为向量。向量是矩阵的一种特例，前面所介绍的有关矩阵的创建及保存的所有方法完全适用于向量。

向量是MATLAB的重要概念之一，它在利用MATLAB进行信号的表示和处理方面发挥着重要的作用（详见第二篇“基于MATLAB的信号与系统分析及实现”）。



在 MATLAB 中，有多种方法可以生成向量，除利用前面已介绍过的创建矩阵的方法来生成向量外，还有以下几种常用方法：

1．利用冒号“：”运算生成向量

冒号运算用于生成等步长（均匀等分）的行向量，其语句格式有以下两种：

- a=m:n
- a=m:p:n

第一种格式用于生成步长值为 1 的均匀等分向量，其中 *m*、*n* 为标量（数量），分别代表向量的起始值和终止值，且 *n*>*m*。例如，输入语句：

```
a=1:10
```

将生成从 1 起始到 10 为止，步长值为 1 的行向量，并赋值给变量 *a*，运行结果为：

```
a =
     1     2     3     4     5     6     7     8     9    10
```

第二种格式用于生成步长值为 *p* 的均匀等分的行向量，其中 *m*、*n* 为标量（数量），分别代表向量的起始值和终止值，*p* 代表向量元素之间步长值，且 *n*>*m*。例如，输入语句

```
b=1:0.1:2
```

将生成从 1 起始到 2 为止，步长值为 0.1 的行向量，并赋值给变量 *b*，运行结果为：

```
b =
  Columns 1 through 7
    1.0000    1.1000    1.2000    1.3000    1.4000    1.5000    1.6000
  Columns 8 through 11
    1.7000    1.8000    1.9000    2.0000
```

2．利用函数 linspace()生成向量

linspace()函数用于生成线性等分向量，其运算规律与冒号运算十分相似，所不同的是该函数除了要给出向量的起始值、终止值以外，不需要给出步长值，而是给出向量元素的个数，其调用格式如下：

- linspace(*m*,*n*)
- linspace(*m*,*n*,*s*)

第一种格式生成从起始值 *m* 开始到终止值 *n* 之间的线性等分的 100 元素的行向量。

第二种格式生成从起始值 *m* 开始到终止值 *n* 之间的 s 个的线性等分点的行向量。例如，

```
linspace(0,10,5)
ans=
         0    2.5000    5.0000    7.5000   10.0000
```

2.4　矩阵的下标

在 MATLAB 系统中，矩阵的元素是通过其行、列的标号来标识的，矩阵元素所处的行号和列号称为该元素的下标。矩阵元素可以通过其下标来引用，***A***(i,j)即表示矩阵 ***A*** 第 i

行第 j 列的元素，例如，若定义矩阵 ***A*** 如下所示：

```
A=[1.1 1.2 1.3
   1.4 1.5 1.6
   1.7 1.8 1.9]
```

则 ***A***(2,3)即是矩阵 ***A*** 第二行第 3 列的元素 1.6。若输入语句：

```
X=A(1,1)+ A(2,2)+ A(3,3)
```

则 ***X*** 即是矩阵 ***A*** 对角线元素之和，运行结果为：

```
X =
4.5000
```

注意

在 MATLAB 中，矩阵下标的行、列号都是从 1 开始的。

我们可以利用矩阵的下标来生成某个矩阵的子阵，命令格式为：

```
B=A(x,y)
```

该命令由已定义的矩阵 ***A***，根据向量 *x* 指定的行和向量 *y* 指定的列生成一新的向量 ***B***。其中 *x* 是由 ***A*** 的行号组成的任意排序的行向量，*y* 是由 ***A*** 的列号组成的任意排序的行向量，*x* 和 *y* 的元素值不应超过 ***A*** 的最大行号和列号。例如，先定义矩阵 ***A*** 为：

```
A=[1  2  3  4
   5  6  7  8
   9  10 11 12
   13 14 15 16]
```

然后，依次输入以下命令：

```
x=[2,4];                    %定义行序号向量
y=[1,3];                    %定义列序号向量
B=A(x,y)
```

上述命令将矩阵 ***A*** 的第 2 和第 4 行、第 1 和第 3 列的元素组成新矩阵并赋值给变量 ***B***。

执行结果为：

```
B =
    5     7
   13    15
```

向量 *x* 和 *y* 中任一个可以是冒号“:”，表示所有的行和列。对上述矩阵 ***A***，***A***(:,y)表示取出矩阵 ***A*** 中 y 所指定位置的所有行，***A***(x,:)表示取出矩阵 ***A*** 中 *x* 所指定位置的所有列。例如：

```
A(:,[1,4])              %取出矩阵 A 所有行的 1、4 列
ans =
     1     4
     5     8
     9    12
```

```
    13    16
 A([2,3],:)                       %取出矩阵 A 的 2、3 行所有列
ans =
     5     6     7     8
     9    10    11    12
```

另外，MATLAB 还可以将若干个小矩阵，通过方括号连接算子，生成较大的矩阵。但各个小矩阵的维数要满足矩阵运算的要求。下面就是将三个小向量（$1\times n$ 的矩阵）构成一个较大向量的实例。

```
a=ones(1,3);              %生成元素值为 1 的行向量
b=zeros(1,4);             %生成 1×4 且元素值为 0 的行向量
c=b;
[b a c]                   %合成较大向量
ans =
     0     0     0     0     1     1     1     0     0     0     0
```

又如

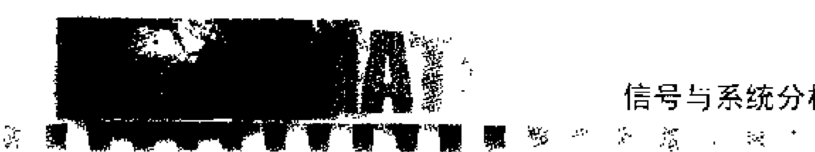
```
a=ones(2,2);
b=a+1;
c=a+2;
d=a+3;
[a b;c d]
ans =
     1     1     2     2
     1     1     2     2
     3     3     4     4
     3     3     4     4
[a b c d]
ans =
     1     1     2     2     3     3     4     4
     1     1     2     2     3     3     4     4
```

2.5 MATLAB 的基本管理命令

1. 工作空间（workspace）的概念及操作

当 MATLAB 启动后，系统自动在内存中开辟一块存储区域用于存储用户在 MATLAB 命令窗口中定义的变量、运算结果和有关数据。此内存空间称为 MATLAB 的工作空间（workspace）。工作空间在 MATLAB 刚启动时为空，此后，用户所定义的变量、运算结果和有关数据均存储在该空间。但用户退出 MATLAB 后，工作空间的内容将不再保留。

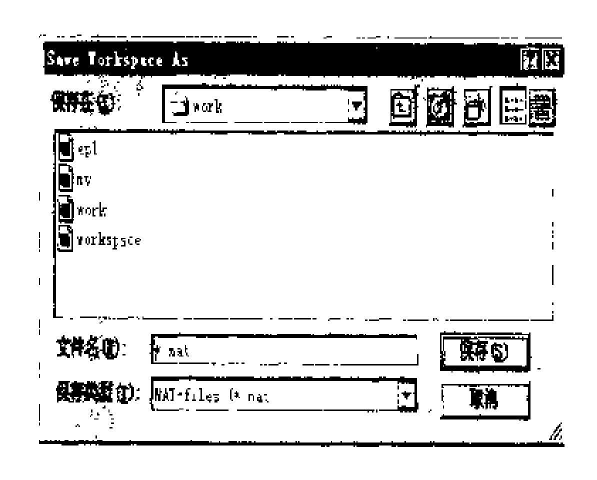
为了能将工作空间的内容长期保留下来，MATLAB 为用户提供了将工作空间以 MAT



文件格式保存到磁盘的功能，具体操作步骤如下。

步骤

（1）保存工作空间。单击 MATLAB 命令窗口菜单栏的文件（【File】）菜单，选择【Save Workspace as】菜单选项，如图 2-1 所示。系统打开工作空间保存设置对话框，如图 2-2 所示。在该对话框中用户可对工作空间保存的路径和文件名进行设置。如输入保存文件名为 workspace.mat，按下【保存】按钮，则系统将工作空间的所有数据以 MAT 文件的格式及文件名 workspace.mat 保存到默认目录 work 中。



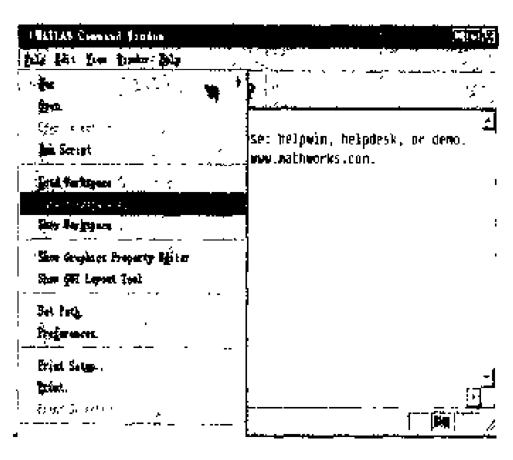


图 2-1　保存工作空间　　　　图 2-2　工作空间保存设置对话框

（2）装载工作空间。在使用 MATLAB 的过程中，如果需要用到上一次已保存的工作空间的变量和数据，可以使用以下操作将已保存的工作空间的变量和数据装载到当前工作空间中。

单击 MATLAB 命令窗口菜单栏的文件（【File】）菜单，选择【Load Workspace】菜单选项，系统打开工作空间装载设置对话框。在该对话框中选定已保存的工作空间的文件名，单击【打开】按钮，即可将已保存的工作空间的内容装载到当前工作空间。

2．清除变量命令 clear

clear 命令用于清除当前工作空间中的指定或全部变量，命令格式为：

```
clear          %清除当前工作空间中的全部变量
clear a b c    %清除当前工作空间中的指定变量 a、b 和 c
```

3．查询变量信息命令 who 的 whos

在使用 MATLAB 的过程中，有时用户需要了解当前工作空间中已赋值的变量的有关信息，这时可使用 who 或 whos 命令。who 命令用于列出工作空间中的变量的清单，whos 命令在列出变量单的同时还给出有关变量的更多信息。下面是使用这两个命令的例子：

```
who
Your variables are:
a        b        r        t
whos
```



```
  Name        Size          Bytes  Class
  a           3×3              72  double array
  b           1×10             80  double array
  r           1×1               8  double array
  t           1×1               8  double array
Grand total is 21 elements using 168 bytes
```

4．其他命令

除上述介绍的命令外，MATLAB 还有几个工作窗口操作的通用命令，如表 2-6 所示

表 2-6　MATLAB 工作窗口操作命令

命　令	功　能
clc	擦除工作窗口中所显示的所有内容
clf	擦除当前图形窗口中的图形
exit	关闭并退出 MATLAB
pack	整理内存碎片以扩大内存空间
cd	改变当前工作目录
dir	列出当前目录及该目录下的文件及子目录清单

5．MATLAB 的行编辑命令

MATLAB 命令是命令行式的交互式命令，用户输入命令后，按下回车键，系统便执行命令并即时给出结果　在命令输入过程中，常用的行编辑键如表 2-7 所示

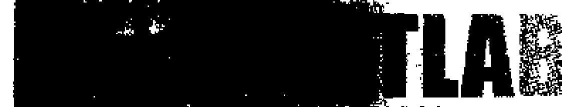
表 2-7　常用行编辑键

键　名	功　能	键　名	功　能
【↑】	前寻式调出已输入过的命令行	【Page Up】	显示上一页
【↓】	后寻式调出已输入过的命令行	【Page Down】	显示下一页
【←】	光标左移一格	【Delete】	删除光标右侧字符
【→】	光标右移一格	【Backspace】	删除光标左侧字符
【Home】	光标移到当前行首	【Esc】	清除当前行的所有内容
【End】	光标移到当前行尾		

在上述编辑键中，最常用的是【↑】和【↓】键，当用户当前准备输入的命令与已执行过的某条命令相似或相同时，即可用【↑】或【↓】键进行前寻或后寻，调出该命令后再进行编辑，可简化输入过程。

2.6　矩阵运算和数组运算

矩阵运算和数组运算是 MATLAB 数值运算的两大运算类型矩阵运算是按矩阵的运算规则进行的，而数组运算则是按数组元素逐一进行的。因此，在进行某些运算（如乘、除）



时，矩阵运算与数组运算有着较大的差别，读者要特别注意。MATLAB 的算术运算符如表 2-2 所示。在 MATLAB 中，可以对矩阵进行数组运算，这时是把矩阵视为数组，运算按数组的运算规则进行。也可以对数组进行矩阵运算，这时是把数组视为矩阵，运算按矩阵的运算规则进行。

2.6.1 矩阵加减与数组加减

矩阵加减与数组加减的运算效果是一致的，运算符也相同，分为下面两种情况。

（1）若参与运算的两矩阵（数组）的维数相同，则加减运算的结果是将两矩阵的对应元素进行加减，例如：

```
A=[1 1 1
   2 2 2
   3 3 3];
B=A;
A+B
ans =
   2   2   2
   4   4   4
   6   6   6
A-B
ans =
   0   0   0
   0   0   0
   0   0   0
```

（2）若参与运算的两矩阵的之一为标量（1×1 的矩阵），则加减运算的结果是将矩阵（数组）的每一元素与该标量逐一相加减，例如

```
A=[1 3 5
   3 1 5
   5 3 1];
A+2
ans =
   3   5   7
   5   3   7
   7   5   3
A-1
ans =
   0   2   4
   2   0   4
   4   2   0
```



2.6.2　矩阵乘与数组乘

矩阵乘与数组乘有着较大差别，运算结果也完全不同。矩阵乘的运算符为“*”，运算是按矩阵的乘法规则进行的，即参与乘运算的两矩阵的内维必须相同。设 ***A***、***B*** 为参与乘运算的两矩阵，***C*** 为 ***A*** 和 ***B*** 矩阵乘的结果，则它们必须满足关系 $\boldsymbol{C}_{m\times n}=\boldsymbol{A}_{m\times k}\boldsymbol{B}_{k\times n}$。因此，参与运算的两矩阵的顺序不能任意调换，因为 ***A**B*** 和 ***B**A*** 计算结果很可能是完全不一样的。例如：

```
a=[1 1 1;2 2 2;3 3 3];
b=a;
a*b
ans =
     6     6     6
    12    12    12
    18    18    18
f=ones(1,3);
g=ones(3,1);
f*g
ans =
     3
g*f
ans =
     1     1     1
     1     1     1
     1     1     1
```

数组乘的运算符为“.*”，运算符中的点号千万不能遗漏，也不能随意加空格符。参与数组乘运算的两数组的大小必须相等（即为同维数组）。数组乘的结果是将两同维数组（矩阵）的对应元素逐一相乘，因此，***A*.*B** 和 ***B*.*A** 计算结果是完全相同的，例如：

```
A=[1 1 1 1 1
   2 2 2 2 2
   3 3 3 3 3];
B=A;
A.*B
ans =
     1     1     1     1     1
     4     4     4     4     4
     9     9     9     9     9
B.*A
ans =
```

```
     1     1     1     1     1
     4     4     4     4     4
     9     9     9     9     9
```

由于矩阵运算和数组运算的差异，能进行数组乘运算的两矩阵，不一定能进行矩阵乘运算。

```
a=ones(1,3);
b=a;
a.*b
ans =
     1     1     1
a*a
??? Error using ==> *
Inner matrix dimensions must agree.
```

2.6.3 矩阵除与数组除

矩阵除与数组除也有着较大差别，运算结果也完全不同。

矩阵除分为矩阵右除和矩阵左除两种情况。矩阵右除的运算符为“/”，设 $\boldsymbol{A}$、$\boldsymbol{B}$ 为两矩阵，则“$\boldsymbol{A/B}$”是指方程 $\boldsymbol{X} * \boldsymbol{B} = \boldsymbol{A}$ 的解矩阵 $\boldsymbol{X}$。显然，矩阵右除运算对参与运算的两矩阵的维数是有一定要求的，即矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 的列数必须相等，例如：

```
a=[1 1 1 1
   2 2 2 2];
b=[1 1 1 1];
x=a/b
x =
     1
     2
x*b
ans =
     1     1     1     1
     2     2     2     2
```

矩阵右除允许参与右除运算的矩阵 $\boldsymbol{B}$ 为标量，这时矩阵右除运算的结果是将矩阵 $\boldsymbol{A}$ 的每一元素逐一与该标量进行除法运算。例如：

```
A=[2 4 6 8
   8 6 4 2];
B=2;
A/B
ans =
     1     2     3     4
```



```
    4    3    2    1
```

矩阵左除的运算符为“\”，设 **A**、**B** 为两矩阵，则“**A\B**”是指方程 $B * X = A$ 的解矩阵 **X**。显然，矩阵右除运算对参与运算的两矩阵的维数是也有一定的要求，即矩阵 **A** 和 **B** 的行数必须相等。

数组右除的运算符为“./”，左除的运算符为“.\”。数组右除和左除的运算结果是完全等效的。设 **A**、**B** 为两同维矩阵，则“**A./B**”的运算结果是将矩阵 **A** 的每一个元素与矩阵 **B** 的对应元素相除。需要就意的是，参与数组运算的两矩阵（数组）的大小必须相等。

```
A=[2 2 3 3 4 4
   1 1 2 2 3 3
   4 4 5 5 6 6];
B=[1 1 3 3 2 2
   1 1 1 1 1 1
   2 2 5 5 3 3];
A./B
ans =
     2     2     1     1     2     2
     1     1     2     2     3     3
     2     2     1     1     2     2
B./A
ans =
    0.5000    0.5000    1.0000    1.0000    0.5000    0.5000
    1.0000    1.0000    0.5000    0.5000    0.3333    0.3333
    0.5000    0.5000    1.0000    1.0000    0.5000    0.5000
```

2.7　MATLAB 的常用矩阵运算函数

1. size()函数

size()函数用于计算矩阵的行数和列数，其调用格式为：

d=size(a)　　将矩阵 a 的行数和列数赋值给变量 d

[m,n]=size(a)　将矩阵 a 和行数赋值给 *m*、列数赋值给 *n*

例如，运行如下命令：

```
a=[1 3 5 7 9
   2 4 6 8 10];
d=size(a)
[m,n]=size(a)
```

运行结果为：

```
d =
5
```

```
m =
     2
n =
     5
```

2．rand()函数

rand()函数用于产生值在 0~1 之间随机分布的随机阵，调用格式为：

rand(*n*)　　产生值在 0~1 之间随机分布的 $n \times n$ 的随机方阵

rand(*m*,*n*)　　产生值在 0~1 之间随机分布的 $n \times m$ 的随机矩阵

例如，运行如下命令：

```
a=rand(4)
b=rand(2,5)
```

运行结果为：

```
a =
    0.9501    0.8913    0.8214    0.9218
    0.2311    0.7621    0.4447    0.7382
    0.6068    0.4565    0.6154    0.1763
0.4860    0.0185    0.7919    0.4057
 b=
    0.3028    0.1509    0.3784    0.8537    0.4966
0.5417    0.6979    0.8600    0.5936    0.8998
```

命令 rand(size(a))则产生维数与矩阵 a 相同的随机矩阵。

3．length()函数

length()函数用于计算矩阵的长度（列数），调用格式为：

a=length(b)　　将矩阵 b 的列数赋值给变量 a

例如，运行如下命令：

```
b=ones(1,100);
a=length(b)
```

运行结果为：

```
a =
   100
```

4．prod()函数

prod()函数用于实现矩阵元素的求积运算，其调用格式为：

- prod(*a*)　若 a 为向量，则该调用格式将计算出矩阵 a 所有元素之积。若 a 为矩阵，则该调用格式将产生一行向量，其元素分别为矩阵 a 的各列元素之积。
- prod(*a*,*k*)　该调用格式将对矩阵 a 按 k 定义的方向进行求积运算，若 k=1 则按列的方向求积，若 k=2 则按行的方向求积。

例如，运行如下命令：



```
a=[1 2 3 4 5];
b=[1 1 1 1
   2 2 2 2
   3 3 3 3];
p1=prod(a)
p2=prod(b)
p3=prod(b,2)
```

运行结果为：

```
p1 =
   120
p2
p2 =
     6     6     6     6
p3
p3 =
     1
    16
    81
```

5. sum()函数

sum()函数用于实现矩阵元素的求和运算，其调用格式为：

- sum(***a***)　若 ***a*** 为向量，则该调用格式将计算出矩阵 ***a*** 所有元素之和。若 ***a*** 为矩阵，则该调用格式将产生一行向量，其元素分别为矩阵 ***a*** 的各列元素之和。
- prod(***a***,***k***)　该调用格式将对矩阵 ***a*** 按 k 定义的方向进行求和运算，若 k=1 则按列的方向求积，若 k=2 则按行的方向求积。

例如，运行如下命令：

```
a=[1 2 3 4 5];
b=[1 1 1 1
   2 2 2 2
   3 3 3 3];
s1=sum(a)
s2=sum(b)
s3=sum(b,2)
s1 =
    15
s2
s2 =
     6     6     6     6
s3 =
```

```
     4
     8
    12
```

6. max()函数

max()函数用于求出矩阵元素的最大值，调用格式为：

- max(*a*)　若 *a* 为向量，则该调用格式将求出向量 *a* 所有元素的最大值　若 *a* 为矩阵，该调用格式将产生一行向量，其元素分别为矩阵 *a* 的各列元素的最大值。
- max(*a*,[ ],*k*)　该调用格式将对矩阵 *a* 按 *k* 定义的方向求最大值，若 *k*=1 则按列的方向求最大值，若 *k*=2 则按行的方向求最大值。

例如，运行如下命令：

```
a=[1 2 3 4 5];
b=[1 1 1 1
   2 2 2 2
   3 3 3 3];
m1=max(a)
m2=max(b)
m3=max(b,[ ],2)
m1 =
     5
m2 =
     3     3     3     3
m3 =
     1
     2
     3
```

另外，与 max()相类似的函数还有 min()（求最小值）和 mean()（求平均值）函数，它们的调用格式与 max 函数完全相同，这里就不一一叙述。

fliplr()函数

fliplr()函数用于实现矩阵的反折运算，即调用格式为：

```
a=fliplr(b)
```

该命令将产生维数与矩阵 *b* 相同的矩阵 *a*，其元素是由矩阵 ***B*** 的元素按列的方向进行反折而得。例如，执行：

```
b=0:10
a=fliplr(b)
```

则结果为：

```
b =
     0     1     2     3     4     5     6     7     8     9    10
a =
```



```
    10     9     8     7     6     5     4     3     2     1     0
```

fliplr函数在生成对称信号（如偶信号）时非常有用。

2.8　关系运算及逻辑运算

在MATLAB中，所有关系运算及逻辑运算都是按数组运算规则定义的。

1．关系运算

MATLAB的基本关系运算符为：

\>（大于）　<（小于）　==（等于）　>=（大于等于）　<=（小于等于）　~=（不等于）

关系运算的规则是：

- 参与关系运算的矩阵必须是同维矩阵或其中之一为标量。
- 当参与运算的矩阵是两同维矩阵 ***A*** 和 ***B*** 时，关系运算的结果是将矩阵 ***A*** 和 ***B*** 下标相同的对应元素逐一进步关系比较，若关系成立则比较结果值为“1”，若关系不成立则比较结果值为“0”。也即关系运算的结果是生成一个与 ***A*** 和 ***B*** 维数相同的矩阵，其元素值为“0”或“1”。
- 当参与运算的矩阵之一为标量时，关系运算的结果是将矩阵的每一个元素与该标量逐一进行关系比较，若关系成立则比较结果值为“1”，若关系不成立则比较结果值为“0”。
- 关系运算比算术运算具有更高的优先权。

```
A=[1 2 3;2 1 3;3 2 1];
B=[2 2 2;2 2 2;2 2 2];
A>=B
ans =
     0     1     1
     1     0     1
     1     1     0
A==2
ans =
     0     1     0
     1     0     0
     0     1     0
```

2．逻辑运算

MATLAB的基本逻辑运算符为：

&（与）　　|（或）　　~（非）

在逻辑运算中，所有非零元素的逻辑值为“真”，用代码“1”表示，值为零的元素的逻辑值为“假”，用代码“0”表示。逻辑运算的规则是：

- 参与逻辑运算的矩阵必须是同维矩阵或其中之一为标量。
- 当参与运算的矩阵是两同维矩阵 ***A*** 和 ***B*** 时，逻辑运算的结果是将矩阵 ***A*** 和 ***B*** 下标相同的对应元素逐一进行逻辑运算，若逻辑运算的值为“真”，则运算结果值为“1”；若逻辑运算的值为“假”，则运算结果值为“0”。即逻辑运算的结果是生成一个与 ***A*** 和 ***B*** 维数相同的矩阵，其元素值为“0”或“1”。
- 当参与运算的矩阵之一为标量时，逻辑运算的结果是将矩阵的每一个元素与该标量逐一进行逻辑运算，若逻辑运算的值为“真”，则运算结果值为“1”；若逻辑运算的值为“假”，则运算结果值为“0”。

三种逻辑运算的真值表如表 2-8 所示。

表 2-8 逻辑运算真值表

a	b	a&b	a|b	~a
1	1	1	1	0
0	1	0	1	1
1	0	0	1	0
0	0	0	0	1

下面是三种逻辑运算的示例。

```
A=[0 1 0 2
   0 3 0 4
   0 5 0 6];
B=[0 1 0 0
   0 1 0 0
   0 1 0 0];
A&B
ans =
     0     1     0     0
     0     1     0     0
     0     1     0     0
A|B
ans =
     0     1     0     1
     0     1     0     1
     0     1     0     1
~A
ans =
     1     0     1     0
     1     0     1     0
     1     0     1     0
```



2.9　MATLAB 的常用数学函数

MATLAB 提供了几乎所有初等函数，包括三角函数、对数函数、指数函数和复数运算函数等。函数的调用格式为：

函数名（变量）

函数的变量即是 MATLAB 的矩阵变量，但函数的运算却是按数组的运算规则进行的，即函数运算的结果，就是将函数运算分别作用于函数变量（矩阵）的每一个元素。表 2-9 列出了 MATLAB 常用初等函数名及其对应功能。

表 2-9　MATLAB 的常用数学函数

函 数 名	功　能	函 数 名	功　能
sin	正弦函数	atanh	反双曲正切函数
cos	余弦函数	abs	求实数绝对值或求复数的模
tan	正切函数	angle	求复数的幅角
asin	反正弦函数	sqrt	平方根函数
acos	反余弦函数	real	求复数的实部
atan	反正切函数	imag	求复数的虚部
sinh	双曲正弦函数	conj	求复数的共轭
cosh	双曲余弦函数	sign	符号函数
tanh	双曲正切函数	exp	自然指数函数（以 e 为底）
asinh	反双曲正弦函数	log	自然对数函数（以 e 为底）
acosh	反双曲余弦函数	log10	以 10 为底的对数函数

下面是应用基本函数的几个例子。

```
a=[1 2 3 4 5 6];                    %定义向量 a
sin(a)                              %计算向量 a 的正弦
ans =
    0.8415    0.9093    0.1411   -0.7568   -0.9589   -0.2794
cos(a)                              %计算向量 a 的余弦
ans =
0.5403   -0.4161   -0.9900   -0.6536    0.2837    0.9602
b=1+2*i                             %定义复数 b
real(b)                             %求复数 b 的实部
ans =
     1
imag(b)                             %求复数 b 的虚部
ans =
     2
```

```
abs(b)                          %求复数 b 的模
ans =
    2.2361
angle(b)                        %求复数 b 的幅角（单位为弧度）
ans =
1071
```

2.10 多项式及其运算

MATLAB 提供了标准多项式的常用函数，包括求根、相乘、相除等，这些功能在进行信号与系统特性分析时非常有用。

1. 多项式的表达与创建

MATLAB 采用将多项式系数按幂次序排列形成的行向量来表征一多项式。设多项式为：

$$A(S)=a_nS^n+a_{n-1}S^{n-1}+\cdots+a_1S+a_0$$

则表征该多项式的行向量为：P=[a_n a_{n-1} ⋯ a_1 a_0]。

因此，在 MATLAB 中，创建多项式即可用创建行向量的方法，直接输入按顺序排序的多项式系数即可。例如，输入语句：

```
A=[1 2 2 1];
```

即表示创建多项式 S^3+2S^2+2S+1，并赋值给变量 A。

2. 多项式求根

函数 roots()用于对多项式求根，调用格式为：

```
p=roots(A)
```

其中 A 为表征多项式的行向量，p 返回该多项式的根（用列向量表示）。例如：

```
B=[1 3 2];                      %创建多项式 S^2 +3S+2
roots(B)                        %求多项式的根
ans =
    -2
-1
A=[1 2 2 1];                    %创建多项式 S^3+2S^2 +2S+1
roots(A)                        %求多项式的根
ans =
  -1.0000
  -0.5000 + 0.8660i
  -0.5000 - 0.8660i
```

3. 由指定根求多项式

函数 ploy 用于由给定根求多项式系数向量，调用格式为：

```
A=poly(p)
```

其中 p 为多项式根（行或列向量表示），A 为返回的多项式系数（行向量表示）。例如

```
p=[2,1];                              %指定多项式的根为 2 和 1
poly(p)                               %求满足指定根的多项式
ans =
     1    -3     2
```

可见 roots()与 poly()互为逆运算。

4. 多项式相乘（卷积）

函数 conv()用于求两个多项式的乘积多项式，其调用格式为：

```
R=conv(A,B)
```

其中 A、B 分别为表征两个多项式的行向量，R 为返回的乘积多项式的系数向量（按降幂次序排列）。例如：

```
A=[1 3 2];                            %创建多项式 S^2 +3S+2
B=[1 2 1];                            %创建多项式 S^2 +2S+1
R=conv(A,B)                           %求多项式 A 与 B 的乘积多项式
R =
     1     5     9     7     2
```

5. 多项式相除（解卷）

函数 deconv()用于进行两个多项式的相除运算，它是相乘运算（conv）的逆运算，其调用格式为：

```
[B,t]=deconv(R,A)
```

其中 R 为被除多项式，A 为除数多项式，B 为商多项式，t 为余多项式。即多项式 R 除以多项式 A 后得于商多项式 B 和余多项式 t。例如：

```
R=[1 5 9 7 2];                 %创建多项式 S^4+5S^3+9S^2 +7S+2
A=[1 3 2];                     %创建多项式 S^2 +3S+2
[B,t]=deconv(R,A)              %求多项式 R 除以多项式 A 后的商多项式 B 和余多项式 t
B =
     1     2     1
t =
     0     0     0     0     0
```

若余多项式系数向量为零向量，则表示 R 能被 A 除尽。



日期：________ 进程：________ 备注：________

第 3 章　MATLAB 程序设计入门





3.1 MATLAB 语言概述

MATLAB 语言体系是 MATLAB 的重要组成部分之一，MATLAB 为用户提供了具有条件控制、函数调用、数据输入输出及面向对象等特性的高层的、完备的编程语言。MATLAB 语言语法简单，程序调试和维护容易，其编程效率远远高于 BASIC、PPSCAL 及 C 等高级语言。

MATLAB 的工作方式有两种，一种是交互式的指令行操作方式，即用户在命令窗口中按 MATLAB 的语法规则输入命令行并按下回车键后，系统将执行该命令并即时给出运算结果。该方式简便易行，非常适合于对简单问题的数学演算、结果分析及测试。但是当要解决的问题变得复杂后，用户将要求系统一次执行多条 MATLAB 语句，显然逐条指令行的交互式人机方式就不再适应大型或复杂问题的解决，这时就要用 MATLAB 的第二种工作方式，即 M 文件的编程工作方式。

M 文件的编程工作方式就是用户通过在命令窗口中调用 M 文件，从而实现一次执行多条 MATLAB 语句的方式。M 文件是由 MATLAB 语句（命令行）构成的 ASCII 码文本文件，即 M 文件中的语句应符合 MATLAB 的语法规则，且文件名必须以.m 为扩展名，如 example.m。用户可以用任何文本编辑器来对 M 文件进行编辑。

M 文件的作用是：当用户在命令窗口中键入已编辑并保存的 M 文件的文件名并按下回车键后，系统将搜索该文件，若该文件存在，系统则将按 M 文件中的语句所规定的计算任务以解释方式逐一执行语句，从而实现用户要求的特定功能。

M 文件又分为命令 M 文件（简称命令文件）和函数 M 文件（简称函数文件）两大类。

3.2 创建、保存与编辑 M 文件

MATLAB 为用户提供了专用的 M 文件编辑器，用来帮助用户完成 M 文件的创建、保存及编辑等工作。

1. 创建新 M 文件

利用 M 文件编辑器创建新 M 文件有如下两种方法：

方法一：启动 MATLAB，选中命令窗口菜单栏【File】菜单下【New】菜单选项的【M-File】命令，打开 MATLAB 的 M 文件编辑器窗口，如图 3-1 所示。

方法二：单击 MATLAB 命令窗口工具栏的"New M-File"图标按钮，也可打开图 3-1 所示的 M 文件编辑器。

在 M 文件编辑器中，用户可以用创建一般文本文件的方法对 M 文件进行输入和编辑。

2. 保存 M 文件

当用户的 M 文件已输入或编辑完毕后，则可将 M 文件保存到磁盘上。保存 M 文件的方法是，单击 M 文件编辑器窗口工具栏中的"Save"图标按钮或选中 M 文件编辑器窗口菜单栏【File】菜单的【Save】命令，打开图 3-2 所示的 M 文件保存对话框。



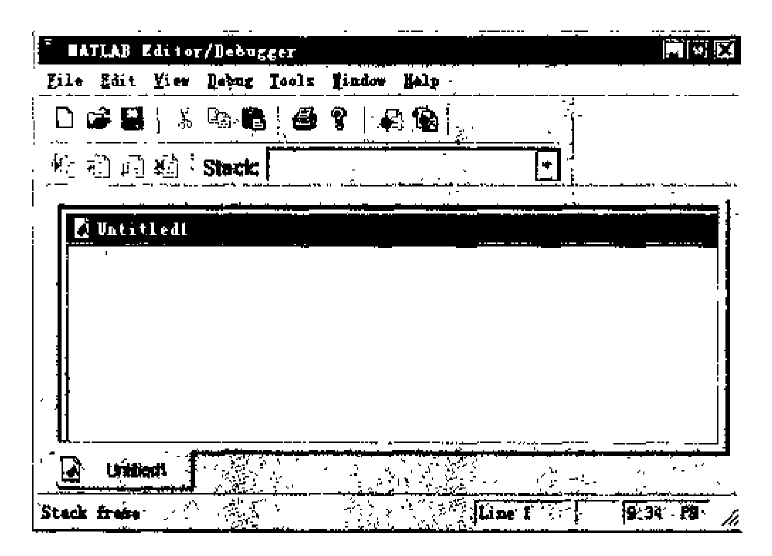


图 3-1　M 文件编辑器窗口

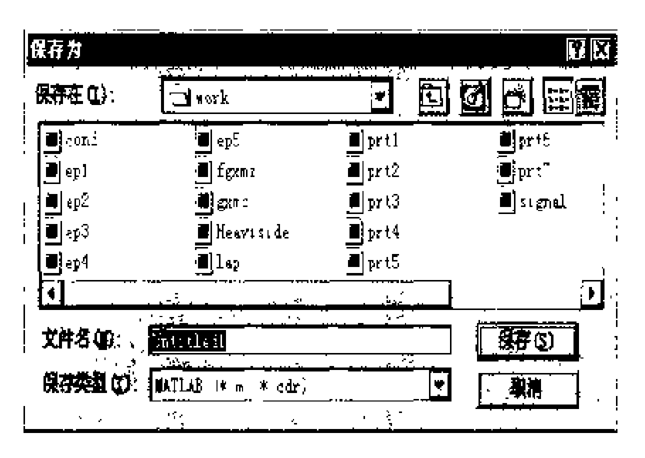


图 3-2　M 文件保存对话框

如果是新建的 M 文件，则系统默认的文件名为“Untitledl.m”，用户可以对要保存的文件进行重新命名。系统默认的文件保存目录为“work”，用户也可以通过保存对话框对文件保存位置进行更改和设置。当保存的文件名和保存位置设定完成后，按下【保存】按钮，即可将 M 文件保存到指定位置。

3．打开 M 文件

若需要对已保存过的 M 文件进行修改和编辑，则可单击 MATLAB 命令窗口工具栏的“Open file”图标按钮或选中命令窗口菜单栏【File】菜单下的【Open】命令，系统即启动 M 文件编辑器并打开用户指定的 M 文件。

4．搜索与执行 M 文件

M 文件中的命令是通过在 MATLAB 命令窗口中键入 M 文件的文件名并按下回车键来执行的。

当用户在命令窗口中键入 M 文件的文件名并按下回车键后，系统先搜索该文件，若该文件存在，则以解释方式按顺序逐条执行 M 文件的语句。




例如，若用户在命令窗口中输入文件名 example 并按回车键后，则系统搜索该文件的过程如下：

（1）在工作空间中寻找变量 example，检查 example 是否为变量。

（2）检查 example 是否为 MATLAB 的内部函数。

（3）在当前目录中搜索文件 example.m。

（4）在 MATLAB 的默认搜索路径中查找文件 example.m。

“work”目录是系统 M 文件的默认保存目录。若用户的 M 文件保存在“work”目录以外的其他位置，则可通过如下两种方式在 MTALAB 命令窗口中直接调用该 M 文件。

方式一：启动 MTALAB 后，用“CD”命令将当前工作目录更改为 M 文件的保存目录，如：

```
cd A:\mydata
```

方式二：用“path”命令将 M 文件的保存目录添加到 MTALAB 的默认搜索路径中。设待执行的 M 文件的保存位置为“D:\myfile”，则添加搜索路径的命令为：

```
path(path, 'D:\myfile')
```

运行该命令后，即可直接在 MTALAB 命令窗口中直接调用并执行 D:\myfile 目录下的所有 M 文件。

3.3 命 令 文 件

命令文件是 M 文件的类型之一，即是由 MTALAB 的语句构成的 ASCII 码文本文件，扩展名为 m。运行命令文件 的效果等价于从 MTALAB 命令窗口中按顺序逐条输入并运行文件是的指令，命令文件具有如下特点：

- 命令文件可以访问 MTALAB 当前工作空间中的所有变量和数据。
- 命令文件运行过程中创建或定义的所有变量均被保留在工作空间中，工作空间中其他命令文件和函数可以共享这些变量。用户可以在命令窗口访问这些变量，并用“who”和“whos”命令对其进行查询。

因此，在程序设计中，命令文件常作为主程序来设计。

3.4 函 数 文 件

函数文件是 M 文件的另一种类型，它也是由 MTALAB 语句构成的 ASCII 码文本文件，扩展名为 m。用户可用前述的 M 文件的创建、保存及编辑的方法来进行函数文件的创建、保存与编辑，但特别需要注意以下几点：

函数文件必须以关键字“function”开头。

函数文件的第一行为函数说明语句，其格式为：

```
function  [返回参数 1, 返回参数 2, …]= 函数名 (传入参数 1, 传入参数 2, …)
```

其中函数名为用户自己定义的函数名（与变量的命名规则相同）。



函数文件保存的文件名应与用户定义的函数名一致，例如，若函数文件说明语句中定义的函数名为“example”，则该函数文件保存的文件名应为“example.m”。

用户可通过函数说明语句中的返回参数及传入参数来实现函数参数传递。返回参数和传入参数并不是必须的。下面是函数文件调用及参数传递的例子。

首先创建如下所示的函数文件并保存。

```
function [m,s]=mean(a)          %定义函数文件mean.m，a为传入参数，m、s为返回参数
l=length(a);                    %计算传入向量长度
s=sum(a);                       %对传入向量求和a并赋值给返回向量s
m=s/l;                          %计算传入向量的平均值并赋值给返回向量m
```

上述函数文件定义了一个新的函数 mean 其作用是对指定向量求和及均值，并通过向量 *s*、*m* 返回计算结果。用户可通过如下所示的命令调用该函数

```
a=1:9;
[s,m]=mean(a)
s =
      5
m =
     45
```

3.5 全局变量和局部变量

用户在命令文件和函数文件中经常都要用到变量，但命令文件中的变量和函数文件中变量却存在着较大的区别。函数内部所定义的变量均为局部变量，它们与其他函数变量是相互隔离的，即变量只在函数内部起作用。而命令文件中变量是全局变量，工作空间的所有命令和函数都可以直接访问这些变量。

当用户需要在多个函数中使用相同的变量时，就要将这些变量定义为全局变量。全局变量的定义由指令“global”实现。例如，语句：

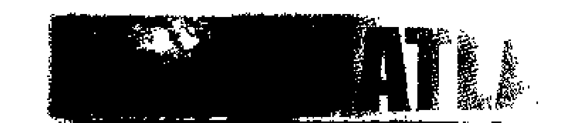
```
global BEG END
```

就定义了两个全局变量“BEG”和“END”。为了不与普通变量相混淆，全局变量通常用大写字母来表示。下面是在函数中如何使用全局变量的例子，首先创建函数文件mean1.m。

```
function s=mean1                %定义函数mean1.m
global BEG END                  %说明全局变量BEG和END
k=BEG:END;                      %由全局变量BEG和END创建向量k
s=sum(k);                       %对向量元素值求和并赋值给返回向量s
```

该函数是一个只有一个返回参数且无传入参数的函数。用户可以通过创建如下所示的命令文件（主程序）来调用该函数。

```
global BEG END              %定义BEG和 END为全局变量
BEG=1;
```




```
END=10;
s1=mean1                    %调用函数 mean1
BEG=1;
END=20;
s2=mean1                    %调用函数 mean1
```

该程序的运行结果为：

```
s1 =
    55
s2 =
        210
```

可见，使用全局变量也可以实现函数参数传递的作用，但这样却破坏了函数的封装性，建议尽量避免使用。

3.6　程序流程控制

MTALAB 为用户提供了丰富的程序结构语句用来实现用户对程序流程的控制。

3.6.1　循环控制语句

当程序中的某段指令需要根据一定的条件多次重复执行时，就需要用到循环控制。在 MATLAB 中，循环控制由 for 语句和 while 语句实现。

1．for 循环语句

for 循环语句的格式为：

```
for  变量=表达式
     语句组
end
```

在上述格式中，end 是必需的，不可默认。表达式是一个矩阵，语句组则是一组合法的 MATLAB 命令。

for 循环语句的运行过程是：从表达式的第一列开始，依次将表达式（矩阵）的各列之值赋值给变量，然后执行语句组中的语句，直到最后一列。

for 语句的典型格式是表达式为冒号运算创建的行向量，即：

```
for  i=m:p:n
     语句组
end
```

其中 m 为循环起始值，n 为循环终止值，p 为步长值。例如，我们用 for 语句来实现求和运算 s=1+3+5+7+…+99，对应的 MATLAB 命令如下：

```
s=0;
for i=1:2:99
```



```
    s=s+i;
end
s
```

运行结果为：

```
s =
        2500
```

在上述格式中，步长 p 可以默认，系统将默认步长为 1，例如，我们可用如下语句来实现求和运算 s=1+2+3+4+…+100

```
s=0;
for i=1:100;
    s=s+i;
end
s
```

运行结果为：

```
s =
        5050
```

for 语句也可实现多重循环的嵌套，其格式为：

```
    for  变量 1=表达式 1
         for  变量 2=表达式 2
         语句组
         end
    end
```

注意

在用 for 语句实现多重循环时，for 和 end 必须成对出现。例如，要定义一个 10×10 的方阵，其每个元素为该元素的行号和列号之和，我们即可用嵌套的二重 for 循环来实现，命令如下：

```
for i=1:10
for j=1:10
    a(i,j)=i+j;
end
end
a
```

运行结果为：

```
a =
     2     3     4     5     6     7     8     9    10    11
     3     4     5     6     7     8     9    10    11    12
     4     5     6     7     8     9    10    11    12    13
     5     6     7     8     9    10    11    12    13    14
```

```
    6     7     8     9    10    11    12    13    14    15
    7     8     9    10    11    12    13    14    15    16
    8     9    10    11    12    13    14    15    16    17
    9    10    11    12    13    14    15    16    17    18
   10    11    12    13    14    15    16    17    18    19
   11    12    13    14    15    16    17    18    19    20
```

2. while 语句

while 语句用来实现在某一逻辑关系控制下的循环。while 语句的格式为：

```
while  关系表达式
          语句组
   end
```

在 while 语句中，end 也是必需的，不可默认。while 语句的执行过程是：首先判断关系表达式是否成立，若成立则运行语句组中的语句，否则停止循环。通常是通过在语句组中对关系表达式进行改变来控制循环是否结束。

例如，我们用 while 语句来实现下列级数求和：

$$s=1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\cdots+\frac{1}{100}$$

命令如下：

```
s=0;
n=1;
while n<=100
  s=s+1/n;
  n=n+1;
end
s
```

运行结果为：

```
s =
     5.1874
```

3.6.2 条件控制语句

和其他高级程序语言一样，MATLAB 也为用户提供的方便的条件控制语句，用以实现程序的条件分支运行。实现条件控制的 MATLAB 命令是 if 语句和 switch 语句。

1. if 语句

if 语句有两种格式。当分支条件只有两种情况时，可采用 if 语句的第一种格式，即：

```
if  表达式
    语句组 1
else
```



```
语句组 2
        end
```

上述 if 语句的运行结果是，如果表达式成立，则运行语句组 1，若表达式不成立，则运行语句组 2。

当程序运行的分支条件多于两个时，则可采用 if 语句的第二种格式，即：

```
if        表达式 1
          语句组 1
elseif    表达式 2
语句组 2
          else
                 语句组 3
          end
```

上述 if 语句的运行结果是，如果表达式 1 成立，则运行语句组 1；若表达式 2 成立，则运行语句组 2；否则，则运行语句组 3。

例如，要定义一个 1×10 的行向量 a，当变量 p 大于 5 时，向量的所有元素值为 0；当变量 p 小于或等于 5 时，向量的所有元素值为 1。我们就可用 if 语句来实现，对应的 MATLAB 命令如下：

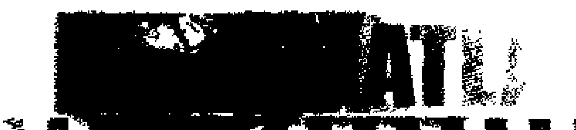
```
if p>5
  a=ones(1,10)
else
  a=zeros(1,10)
end
```

2. switch 语句

当程序运行过程中需要根据某个变量的多种不同取值情况来运行不同的语句时，就要用到 switch 语句。

switch 语句适合于多个条件的分支程序，其基本格式为：

```
switch   控制变量
case     变量值 1
语句组 1
case     变量值 2
语句组 2
case     变量值 3
语句组 3
 ·
 ·
 ·
otherwise
语句组 n
```



end

在上述格式中，switch 语句通过判断控制变量的取值情况来决定运行哪一个语句组，即当控制变量的值为变量值 1 时，则运行语句组 1；当控制变量的值为变量值 2 时，则运行语句组 2，依此类推。若所有条件均不满足，则运行 otherwise 后的语句组。

注意，在 switch 语句中，end 是必须的，不可默认。

例如，我们可以用 switch 语句来实现根据变量 k 的不同取值情况，从而将当前图形窗口的背景设置为不同的颜色，命令如下：

```
switch   k
        case 1
             set(gcf,'color','r')
        case 2
             set(gcf,'color','w')
        case 3
             set(gcf,'color','y')
        case 4
             set(gcf,'color','b')
        otherwise
             set(gcf,'color','g')
  end
```

第 4 章　MATLAB 的符号运算功能





除了数值计算以外，在数学、工程和其他应用科学中还经常用到符号运算。MATLAB 和著名的符号运算语言 Maple 结合，为用户提供了集符号运算与符号可视化为一体的强大的符号运算功能。

4.1 符号对象的创建和使用

在数值计算过程中，参与运算的变量都是被赋了值的数值变量。而在符号运算的整个过程中，参与运算的是符号变量，在符号运算中所出现的数字都是当做符号来处理的。

在 MATLAB 的 Symbolic Math Toolbox 中定义了一种新的数据类型为符号对象（Symbolic object）或 sym。符号对象是用来存储代表符号的字符串，代表了符号变量、符号表达式和符号矩阵。

4.1.1 符号运算入门

符号运算的特点是，运算过程中允许存在非数值的符号变量。让我们先看下面的示例，看看符号运算在数学运算中的运用。

例如，对于函数 $f(x)=(\sin x)^2$，用 MATLAB 求它的微积分，命令如下：

```
f='sin(x)^2';          %定义符号函数 f(x)
dfdx=diff(f)           %求 d/dx f(x) 的指令
intf=int(f)            %求 ∫f(x)dx 的指令
```

显示的计算结果为：

```
dfdx=
2*sin(x)*cos(x)
intf=
-1/2*sin(x)*cos(x)+1/2*x
```

所以，$\dfrac{\mathrm{d}f(x)}{\mathrm{d}x}=2\sin x\cos x$，$\int f(x)\mathrm{d}x=\dfrac{1}{2}x-\dfrac{1}{2}\sin x\cos x$。在这个例子中，我们首先定义符号函数 f(x)='sin(x)^2'，然后由符号运算获得 $f(x)=(\sin x)^2$ 的微分和积分。下面我们就介绍如何用 MATLAB 来实现符号运算及符号运算可视化。

4.1.2 定义符号变量

在使用符号变量之前，应先声明某些要用到的变量是“符号”变量。声明符号变量的语句是：

```
syms 变量名列表
```

其中各个变量名应该用空格分隔，而不能用逗号分隔。

或者用：sym('变量名') 来创建符号变量。

例如，下面的例子创建了符号变量 *x* 和 *a*：

```
x=sym('x')
a=sym('alpha')
```

或者使用：syms *x a* 定义符号变量 *x* 和 *a*。

这里，变量 *x* 和 *a* 的类型是符号对象，它们被定义后，就可以参与符号运算。

4.1.3　定义符号表达式和符号方程

符号表达式和符号方程是两种不同的操作对象。它们的区别在于：符号表达式不包含等号（=），而符号方程必须带等号。但它们的创建方式是相同的。

例如：要考虑二次函数 *f*=a*x*^2+b*x*+c。可以创建符号表达式，赋值给符号变量 *f*。

f=sym('a**x*^2+b**x*+c');

或者，*f*='a**x*^2+b**x*+c';

在这个例子中，将符号表达式赋给符号变量 *f*，但这不是必需的，引入符号变量是为了以后调用方便。

在这种情况下，没用创建对应于表达式中 *a*、*b*、*c*、*x* 项的变量，为了执行符号数学运算（如微分、积分等），必须显式地创建这些变量，可以用以下命令创建：

```
syms a b c x
```

例如，下面的例子创建了符号表达式和符号方程，分别赋值给相应的符号对象。

```
sym x a b c
f='sin(x)^2';                %创建符号表达式 sin(x)^2 赋给变量 f
eq='a*x^2+b*x+c=0'           %创建的符号方程赋给变量 eq
```

4.1.4　定义抽象函数和符号数学函数

如果要创建抽象函数 *f*(*x*)，可以使用：

f=sym('f(*x*)')

则 *f* 就像 *f*(*x*) 一样参与运算。

在 Symbolic Math Toolbox 中可利用符号表达式创建符号数学函数。

例如，下面的命令将产生符号表达式 *r*，*t* 和 *f*：

```
syms x y z
r=sqrt(x^2+y^2+z^2)   或者 r=sym('sqrt(x^2+y^2+z^2)')
t=atan(y/x)
f=sin(x*y)/(x*y)
```

4.2 数值与符号的转换

sym 函数有四种选项将数值结果转换为符号表达式。

例如，对于变量 *rho*，定义为：$rho=\dfrac{1+\sqrt{5}}{2}$，在 MATLAB 的命令窗口中定义 *rho*：

```
rho=(1+sqrt(5))/2;
```

将这个数值结果转化为符号表达式：

- sym(rho,'f');

返回符号浮点表示形式。结果为：

```
sym(rho,'f')
ans =
'1.9e3779b97f4a8'*2^(0)
```

- sym(rho,'r');

返回符号有理数表示形式，这是 sym 的默认设置，当调用 sym 而没有第二个参数时，相当于 sym 使用了 r 选项。结果为：

```
sym(rho,'r')
ans =
7286977268806824*2^(-52)
```

例如：

```
sym(0.75)
sym(0.75)
ans =
3/4
```

- sym(rho,'e');

返回符号有理数表示形式。同时根据 eps 给出 rho 的理论表达式和实际计算的差。

```
sym(rho,'e')
ans =
7286977268806824*2^(-52)
```

这里，rho 的理论值和实际浮点值相同，对于 1/3，可得：

```
sym(1/3,'e')
ans =
1/3-eps/12
```

- sym(rho,'d');

返回符号十进制小数表示形式。有效位数由 digits 定义。digits 的默认值是 32。

```
sym(rho,'d')
ans =
1.6180339887498949025257388711907
```

如果想要一个较短的结果，可由 digits 定义有效位数。

```
digits(5)
sym(rho,'d')
ans =
1.6180
```

在上面的四种格式中，最后得到的都是符号对象变量。

4.3　符号算术运算

在 MATLAB 的数值计算中，加、减、乘、除等运算很简单直观。但在符号运算中，情况就不同了，所有涉及符号的运算都要借助专用函数进行，下面这些符号运算指令，适用于符号表达式和符号矩阵。

4.3.1　定义符号矩阵

MATLAB 中的数值矩阵不能直接参与符号运算，必须经过转换。sym 函数的一个非常有用的功能就是将数值矩阵转换成符号矩阵。

例如，下面的代码将产生一个 3×3 的 **Hibert** 矩阵。

```
A=hilb(3)
A =
     1.0000     0.5000     0.3333
     0.5000     0.3333     0.2500
     0.3333     0.2500     0.2000
```

对 ***A*** 使用 sym 命令可以获得一个具有无穷精度的 3×3 的符号形式的矩阵。

```
A=sym(A)
A =
[   1, 1/2, 1/3]
[ 1/2, 1/3, 1/4]
[ 1/3, 1/4, 1/5]
```

当调用 Symbolic Math Toolbox 时，默认地调用有理算术运算。当使用 sym 函数创建符号变量时，有理算术运算将被启动。

例如，对于双精度矩阵 ***A***，sym 函数将转化为对应的符号矩阵形式。

```
A =
     1.1000     1.2000     1.3000
     2.1000     2.2000     2.3000
     3.1000     3.2000     3.3000
A=sym(A)
A =
```

```
[ 11/10,   6/5, 13/10]
[ 21/10,  11/5, 23/10]
[ 31/10,  16/5, 33/10]
```

对于非数值的符号矩阵，通过 sym 函数，可用与普通矩阵定义类似的格式来定义。

例如，下面的例子将定义符号矩阵 ***A***、符号矩阵 ***B*** 和符号矩阵 ***C***。

```
syms a b c d x
A=sym('[a b;c d]')
A =
[ a, b]
[ c, d]
B=sym('[2*a+b,3*b;5*c+d,2*d]')
B =
[ 2*a+b,    3*b]
[ 5*c+d,    2*d]
C=sym('[1/(1+a),sin(x),(b-x)/(a+x);1,exp(x),x^2]')
C =
[1/(1+a),sin(x), (b-x)/(a+x)]
[1,exp(x),x^2]
```

4.3.2 符号矩阵的加、减运算

实现符号矩阵加（add）、减（subtract）的指令是：

```
symadd(A,B)              %给出两个符号矩阵的和 A+B;
symsub(A,B)              %给出两个符号矩阵的差 A-B;
```

显然，参与符号加、减运算的符号矩阵应具有相同的维数。

例如，对于上面定义的符号矩阵 ***A*** 和 ***B***，利用 symadd 和 symsub 的结果是：

```
symadd(A,B)
ans =
[ 3*a+b,    4*b]
[ 6*c+d,    3*d]
symsub(A,B)
ans =
[    -a-b,    -2*b]
[ -4*c-d,       -d]
```

对于具有数值的符号矩阵和加法，使用同样计算方法。

例如，下面的例子计算两个符号矩阵的和。

```
A=hilb(3)
    1.0000    0.5000    0.3333
    0.5000    0.3333    0.2500
```



```
    0.3333    0.2500    0.2000
A=sym(A)
A =
[   1, 1/2, 1/3]
[ 1/2, 1/3, 1/4]
[ 1/3, 1/4, 1/5]
B=sym('[2*a,b,c;a,2*b,c;a,b,2*c]')
B =
[ 2*a,   b,   c]
[   a, 2*b,   c]
[   a,   b, 2*c]
symadd(A,B)
ans =
[   1+2*a,   1/2+b,   1/3+c]
[   1/2+a, 1/3+2*b,   1/4+c]
[   1/3+a,   1/4+b, 1/5+2*c]
```

4.3.3 符号矩阵的乘、除运算

实现符号矩阵乘、除的指令是：

```
symmul(A,B)          %给出两个符号矩阵的积 A*B；
symdiv(A,B)          %给出两个符号矩阵的商 A/B；
```

例如，对于上面定义的符号矩阵 ***A*** 和 ***B***，利用 symmul 和 symdiv 的结果是：

```
symmul(A,B)
ans =
[ a*(2*a+b)+b*(5*c+d),        3*a*b+2*b*d]
[ c*(2*a+b)+d*(5*c+d),        3*c*b+2*d^2]
symdiv(A,B)
ans =
[(2*a*d-b*d-5*c*b)/(-15*c*b-b*d+4*a*d),        -b*(-b+a)/(-15*c*b-b*d+4*a*d)]
[        -d*(3*c+d)/(-15*c*b-b*d+4*a*d), (2*a*d-3*c*b+b*d)/(-15*c*b-b*d+4*a*d)]
```

4.3.4 符号变量替换

在很多时候，当我们得到一个符号解或者在一个符号表达式中，需要将一些符号变量替换成数字或其他符号。这在 MATLAB 的符号运算中可利用 subs 函数实现。

subs 函数适用于单个符号矩阵、符号表达式、符号代数方程和微分方程。subs 函数有如下两种格式：

（1）subs(S,NEW)用新变量 NEW 替换 S 中的默认变量。

例如，对于下面的符号矩阵 ***G***，用'pi/3'替换 ***G*** 中的默认变量 x。

```
G=sym('[a*sin(b+x),a+b,exp(a*x),sqrt(x)]');
G1=subs(G,'pi/3')
G1 =
[ a*sin(b+pi/3),          a+b,    exp(a*pi/3),       sqrt(pi/3)]
```

和用 pi/3 替换 G 中的默认变量 x 比较：

```
G=sym('[a*sin(b+x),a+b,exp(a*x),sqrt(x)]');
G1=subs(G,pi/3)
```

结果为：

```
G1 =
[a*sin(b+1/3*pi),          a+b,         exp(1/3*a*pi), 1/3*3^(1/2)*pi^(1/2)]
```

（2）subs(S,NEW,OLD)　用新变量 NEW 代替 S 中的指定变量 OLD。

例如，对于下面的符号表达式 f，用 2 替换 f 中的默认变量 a。

```
f=sym('sin(1/3*a*pi)');
subs(f,'2','a')
```

结果为：

```
ans =
sin(1/3*2*pi)
```

如果用数值替换符号表达式中的变量，将得到相应变量由数值替代的表达式或数值。

```
f=sym('sin(1/3*a*pi)');
subs(f,2,'a')
```

结果为：

```
ans =
    0.8660
```

4.4　符号微积分运算

4.4.1　确定符号变量

当进行数学运算时，对应变量的选取很容易得到。例如，对于函数 $f = x^n$，当我们对 f 求导数时，自然地是对 x 求导，n 看成参数（常数）。而在 MATLAB 中，如何知道是对 x 求导而不是对 n 求导呢？它是通过符号表达式中隐含的符号变量来确定的。

在 Symbolic Math Toolboxs 中，确定一个符号表达式中的符号变量的规则是：

（1）只对（除 i，j 之外的）单个小写英文字母进行检索。

（2）小写字母 x 是首选符号变量。

（3）其余小写字母被选为符号变量的次序：在英文字母表中，靠近“x”的优先，在



"x" 之后的优先。

按照这个规则，对 $f = x^n$ 求导数时，自然是对 x 求导，n 看成参数（常数）。

工具箱还提供了 findsym 函数来确定表达式中的符号变量。例如，findsym(*f*,1)寻找第一个符号变量。findsym 函数中的第二个参数，代表了在符号对象中想要寻找的符号变量的个数，如果默认，将给出符号表达式中的所有符号变量。

例如，下面的示例代码给出了 f 中的符号变量。

```
syms a b c x
f=sym('a*x^2+b*x+c');
findsym(f,1)
ans =
x
findsym(f,2)
ans =
x,c
```

4.4.2　符号微分运算

MATLAB 中，对符号表达式微分的函数是 diff()。利用这个函数，可以求符号表达式的一阶导数、n 阶导数。该函数有如下三种调用格式：

1．diff(*f*)

对符号表达式 f 进行微分运算，符号变量由 4.4.1 小节的规则决定。

例如，在下面的命令中，可求出 f 关于变量 x 的微分。

```
syms a x;
f=sin(a*x);
df=diff(f)
```

或者输入：

```
f='sin(a*x)'
df=diff(f)
```

结果显示为：

```
df=
cos(a*x)*a
```

2．diff(*f*,a)

符号表达式 f 对变量 a 进行微分运算。

例如，在下面的命令中，分别计算 f 对变量 x 和 n 微分。

```
syms x n
f=x^n
diff(f,x)
ans =
```

```
x^n*n/x
diff(f,n)
ans =
x^n*log(x)
```

3. diff(f,n)或 diff(f,a,n)

计算 f 对默认变量或指定变量的 n 阶导数。这里 n 是一个具体的数值（正整数），用于指定求导的阶数。

例如，在下面的代码中，计算 f 关于变量 x 的 2 阶导数。

```
syms a x;
f=sin(a*x);
df=diff(f,x,2)
df=
-sin(a*x)*a^2
```

diff 函数也可以使用符号矩阵作为它的输入，在这种情况下，微分按矩阵元素逐个进行。

4.4.3 符号积分运算

MATLAB 的符号工具箱还提供了用于积分运算的 int()函数。该函数也有三种调用格式：

1. int(s)

对于符号变量 s 代表的符号表达式，求 s 的不定积分，积分变量由 4.4.1 小节的规则确定。

例如，对于下面的代码，求 s 关于变量 x 的不定积分。

```
syms x;
s='1/(1+x^2)';
f=int(f)
```

结果为：

```
f=
atan(x)
```

2. int(s,v)

计算 s 关于变量 v 的不定积分。

例如，下面的例子将对指定的变量求不定积分。

```
syms alpha;
s='sin(alpha*u)';
int(s,alpha)
ans=
-1/u*cos(alpha*u)
```

3．int(s,a,b) 或 int(s,v,a,b)

计算 s 关于变量 x 或 v，从 a 到 b 的定积分。

例如，下面的例子将对指定的变量求从 2 到 sin(t)的定积分。

```
syms x t;
s=4*x*t;
int(s,2,sin(t))
```

或者：int(s,x,2,sin(t))

结果为：

```
ans=
2*t*(sin(t)^2-4)
```

例如，下面的例子将计算：$x1$*log(1+$x1$)关于 $x1$ 求从 0 到 1 的定积分。

```
syms x1;
int(x1*log(1+x1),0,1)
ans=
1/4
```

4.4.4　符号微积分运算示例

对于微分和积分运算，在信号处理中运用较多，下面对微分和积分运算再举几个例子。

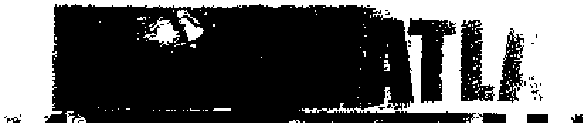
例 4-1：对于函数 $f(x,y)=x\sin y$，求 $\dfrac{\partial f(x,y)}{\partial x\partial y}$。

```
syms x y;
f='x*sin(y)';
dfdxdy=diff(diff(f,x),y)
dfdxdy=
cos(y)
```

例 4-2：对于函数 $s(x,y)=xe^{-xy}$，先求 s 关于 x 的不定积分，再求所得结果关于 y 的不定积分，即计算 $\int\left(\int xe^{-xy}\,\mathrm{d}\,x\right)\mathrm{d}\,y$。

```
syms x y
s='x*exp(-x*y)'
int(int(s,x),y);
ans=
1/y*exp(-x*y)
```

即　$\int\left(\int xe^{-xy}\,\mathrm{d}\,x\right)\mathrm{d}\,y=\dfrac{e^{-xy}}{y}$。



例 4-3：给定一个函数：$\frac{1}{x^2+4x+3}\sin x$

下面的代码将对这个函数求微分。

```
syms x y y1 y2;
y='1/(x^2+4*x+3)*sin(x)';
y1=diff(y,x)
y1=
-1/(x^2+4*x+3)^2*sin(x)*(2*x+4)+1/(x^2+4*x+3)*cos(x)
```

对得出的微分结果再进行积分运算，得出了另一个表达式。

```
y2=int(y1,x)
y2 =
-1/2*sin(x)/(x+3)+1/2*sin(x)/(x+1)
```

这个表达式和原函数并不一样，将这个表达式进行化简处理后，则可以得出和原函数一致的结果。

```
y2=simplify(y2)
y2 =
1/(x^2+4*x+3)*sin(x)
```

或者使用 simple 函数，进行化简：

```
y2=simple(y2)
y2 =
sin(x)/(x+3)/(x+1)
```

通常，用户希望用最简单的形式书写表达式。MATLAB 提供了用于化简符号表达式的函数，从而使表达式变得较为简单。

例如，使用 simple(f)化简表达式：

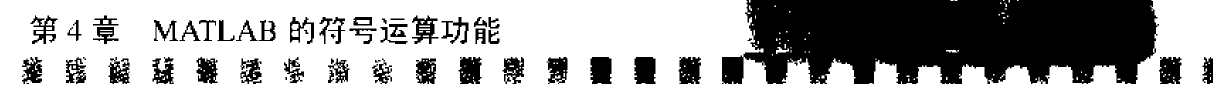
```
syms x;
f=sym('cos(x)^2+sin(x)^2');
f=simple(f)
f =
1
```

simplify 函数是一个普遍使用的强有力的表达式化简工具，它可以对包含和式、根式、分数、乘方、指数、对数、三角函数等的表达式进行化简。

此外，simple 函数也是一个常用的化简表达式的函数。simple 函数通常是寻找表达式的化简形式，使之包含最少的字符。它将 simplify 函数和其他一些化简函数应用于表达式的结果记下，最后 simple 给出最短的结果。

在上面这个示例中，通过符号微分和积分后得到和原函数一致的结果，但在 MATLAB 的符号微积分运算中，常常会有运算后结果不一致的情况。例如，在下面的示例中，得到的结果和原函数不一样，但它们的差是一个常数。

为了直观起见，我们使用下一节将要介绍的 ezplot 符号绘图函数，在图形窗口绘制出微分和积分运算的结果，关于绘图处理，请参看 4.5 节和第 5 章。



例如，在下面的示例中，创建一个符号函数 $f(x)=\dfrac{1}{5+4\cos x}$。

```
x=sym('x');
f=sym('1/(5+4*cos(x))');
ezplot(f);
```

利用 ezplot 函数在图形窗口中默认的绘图区域绘制 f 的曲线，如图 4-1 所示。

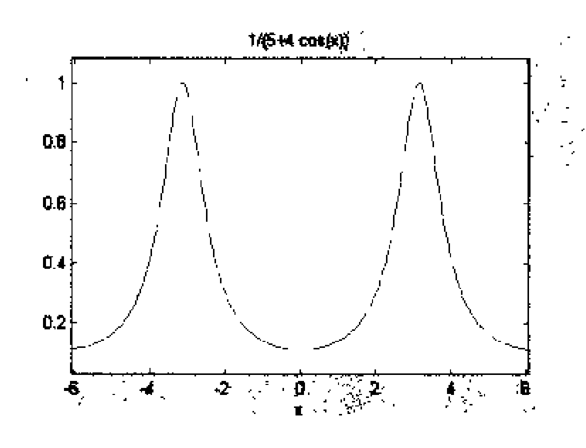


图 4-1 f 的图形

对 f 进行微分操作：

```
f1=diff(f)
f1 =
4/(5+4*cos(x))^2*sin(x)
ezplot(f1)
```

在图形窗口绘制出 f1 的图形，如图 4-2 所示。

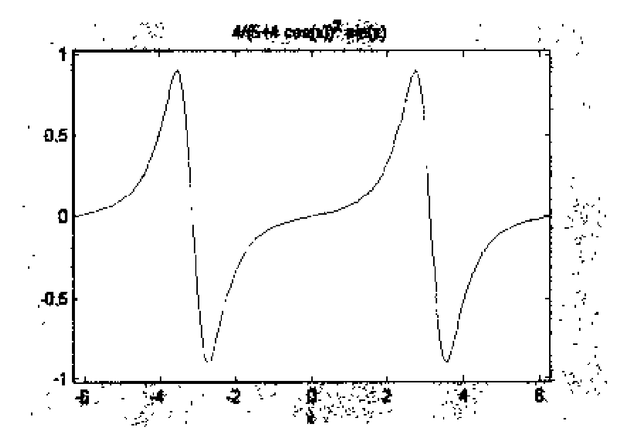


图 4-2 f 微分后的图形

```
f2=diff(f,2)
f2 =
32/(5+4*cos(x))^3*sin(x)^2+4/(5+4*cos(x))^2*cos(x)
ezplot(f2)
```

在图形窗口绘制出 f2 的图形，如图 4-3 所示。

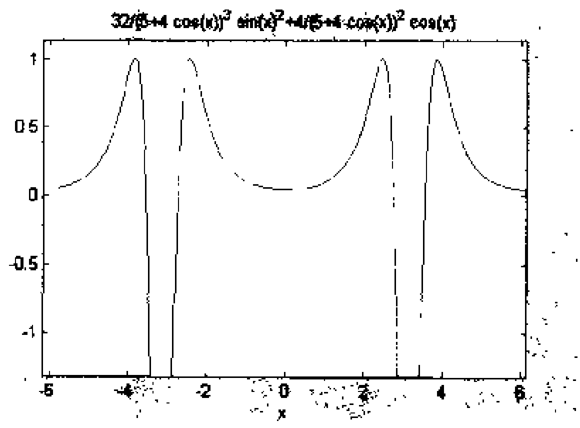


图 4-3　f 两次微分后的图形

```
g=int(int(f2))
g =
-8/(tan(1/2*x)^2+9)
```

在图形窗口绘制出 g 的图形，如图 4-4 所示。

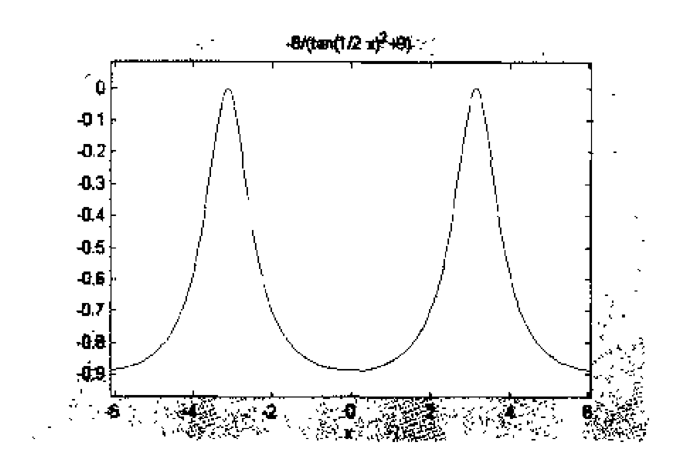


图 4-4　f 两次微分和重积分后的图形

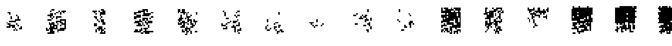
我们发现，g 和原函数形式不相同，但图形相似。

```
e=simple(f-g)
e =
1/(5+4*cos(x))+8/(tan(1/2*x)^2+9)
ezplot(e)
```

此时，在图形窗口中绘制的 f-g 的图形是一条直线。对 e 进行化简：

```
e=simple(e)
e =
1
```

所以 f-g 确实是一个恒定常数。



4.5　符号函数的可视化

4.5.1　绘制二维符号函数曲线

对于符号函数，MATLAB 提供了一个非常简单的作图指令：ezplot()函数。通过这个命令，可以在图形窗口绘制出符号函数的图形。ezplot()函数有如下几种调用格式。

- ezplot(f)

对于符号函数 f=f(x)，按照 x 默认的范围：$-2*pi<x<2*pi$，在图形窗口中绘制出 f=f(x) 的图形。

例如，在 MATLAB 的命令窗口中输入如下代码，将得到如图 4-5 所示的图形。

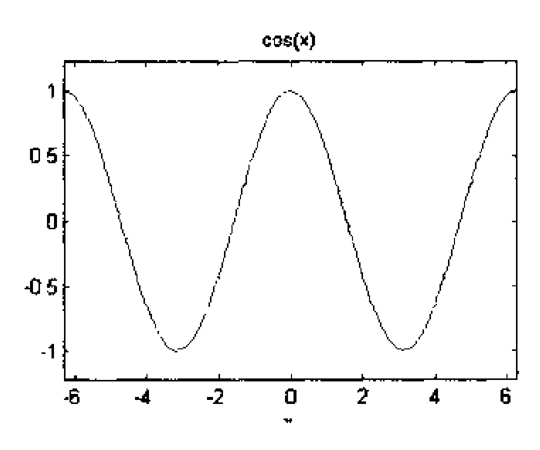


图 4-5　ezplot(f)绘制的图形

```
f='cos(x)';
ezplot(f);
```

- ezplot(f,[a,b])

在图形窗口中绘制符号函数 f=f(x)的图形，x 的范围由[a,b]确定，即 $a<x<b$。

例如：在 MATLAB 的命令窗口中输入如下代码，将得到如图 4-6 所示的图形。

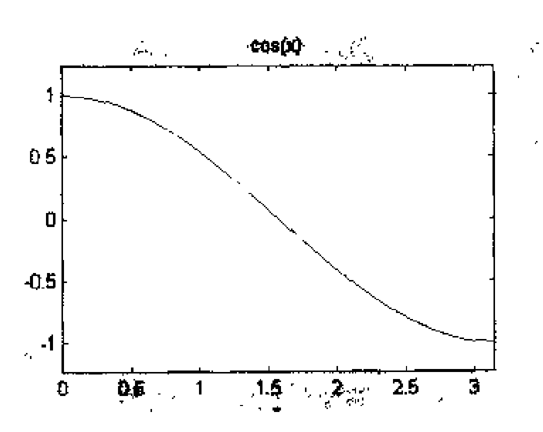


图 4-6　ezplot(f,[a,b])绘制的图形




```
f='cos(x)';
ezplot(f,[0,pi]);
```

- ezplot(f)

对于符号函数 f=f(x,y)，ezplot(f)在图形窗口中绘制符号方程 f(x,y)=0 的图形。x 和 y 的取值范围为：

-2*pi<x<2*pi，-2*pi<y<2*pi

例如，在 MATLAB 的命令窗口中输入如下代码，将得到如图 4-7 所示的图形。

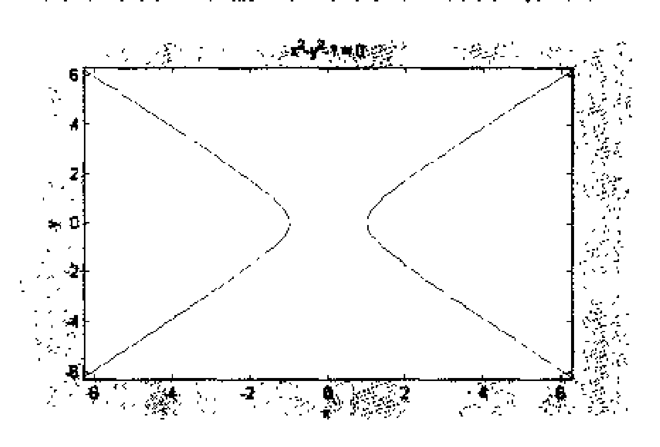


图 4-7 绘制的符号方程的图形

```
f='x^2-y^2-1';
ezplot(f);
```

- ezplot(f,[xmin,xmax,ymin,ymax])

对于符号函数 f=f(x,y)，ezplot(f,[xmin,xmax,ymin,ymax])在图形窗口中绘制符号方程 f(x,y)=0 的图形。x 和 y 的取值范围为：

xmin<x<xmax，ymin<ymax

当没有指定 y 的范围时，y 的范围和 x 的一致。

例如，在 MATLAB 的命令窗口中输入如下代码，将得到如图 4-8 所示的图形。

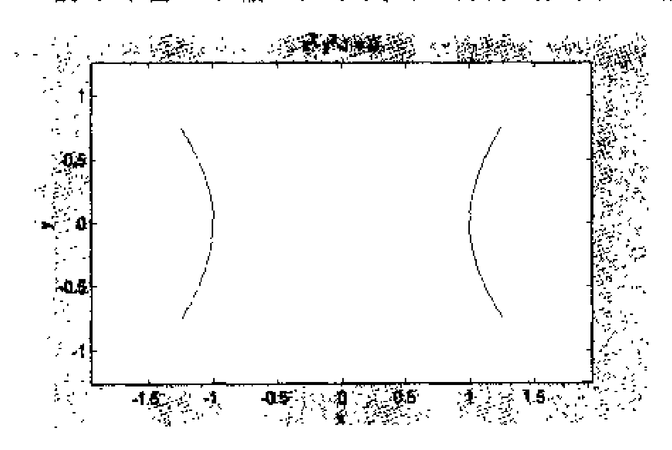
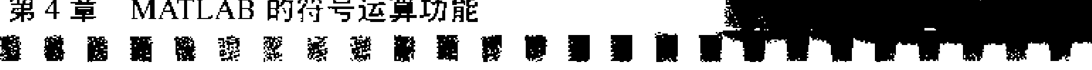


图 4-8 指定范围的符号方程图形

```
f='x^2-y^2-1';
ezplot(f,[-1.25,1.25]);
```



```
axis equal;
```

- ezplot(*x*,*y*)

对于符号函数 $x=x(t)$，$y=y(t)$，ezplot(*x*,*y*)在图形窗口中绘制符号方程 $x=x(t)$、$y=y(t)$的图形。t 的取值范围为：$0<t<2*pi$

例如，在 MATLAB 的命令窗口中输入如下代码，将得到如图 4-9 所示的图形。

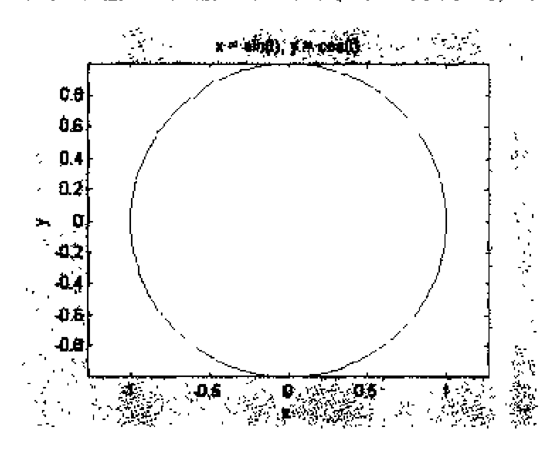


图 4-9　$x=x(t)$，$y=y(t)$的图形

```
x='sin(t) ';
y='cos(t) ';
ezplot(x,y);
```

- ezplot(x,y,[tmin,tmax])

对于符号函数 $x=x(t)$，$y=y(t)$，ezplot(*x*,*y*)在图形窗口中绘制符号方程 $x=x(t)$，$y=y(t)$的图形。t 的取值范围为：tmin<t<tmax

例如，在 MATLAB 的命令窗口中输入如下代码，将得到如图 4-10 所示的图形。

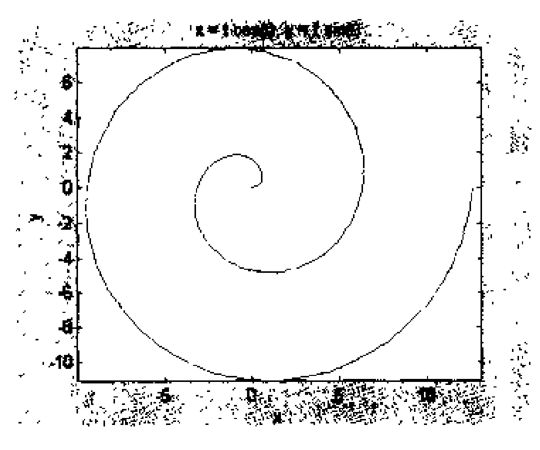


图 4-10　指定范围的 $x=x(t)$，$y=y(t)$图形

```
x='t*cos(t)';
y='t*sin(t)';
ezplot(x,y,[0,4*pi]);
```

对于上面所列的几种绘图格式，也可以是：

ezplot(f,fig)、

ezplot(f,[a,b]),fig)、

ezplot(f,[xmin,xmax,ymin,ymax],fig)

或 ezplot(x,y,[tmin,tmax],fig)

这些格式将在由 fig 指定的图形窗口中绘制图形。

例如，在 MATLAB 的命令窗口中输入如下代码，将在新的图形窗口 Figure No.8 中得到如图 4-11 所示的图形。

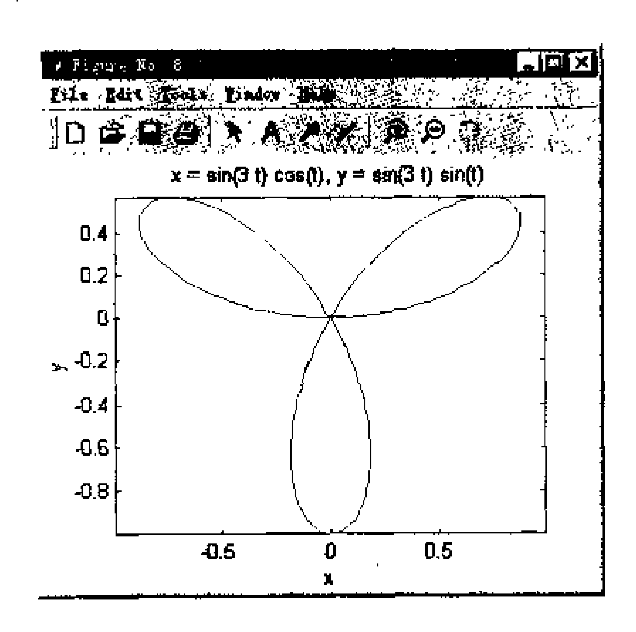


图 4-11 在新窗口中绘制的图形

```
x='sin(3*t)*cos(t)';
y='sin(3*t)*sin(t)';
ezplot(x,y,[0,pi],8);
```

4.5.2 绘制三维符号函数曲线

对于符号函数，MATLAB 还提供了用于三维空间的作图函数：ezplot3()函数和 ezsurf()函数。通过这两个命令，可以在图形窗口绘制出三维符号函数的图形。

1．ezplot3()函数的使用

- ezplot3(x,y,z)

对于符号函数 $x=x(t)$，$y=y(t)$，$z=z(t)$，ezplot3(x,y,z)在图形窗口中绘制符号方程 $x=x(t)$，$y=y(t)$，$z=z(t)$的图形。t 的取值范围为默认值：$0<t<2*pi$。

例如，在 MATLAB 的命令窗口中输入如下代码，将得到如图 4-12 所示的图形。



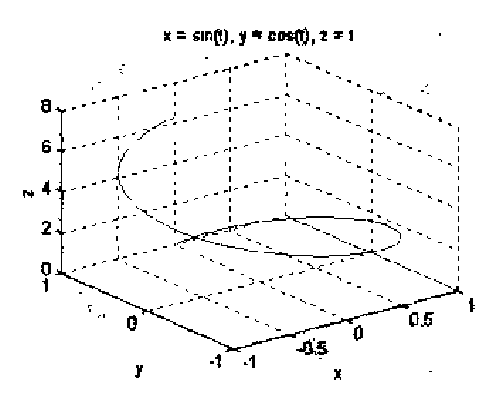


图 4-12　利用 ezplot3(x,y,z)绘制的图形

```
x='sin(t) ';
y='cos(t) ';
z='t';
ezplot3(x,y,z)
```

也可直接输入：

ezplot3('sin(t) ', 'cos(t) ', 't')来绘制图形。

- ezplot3(x,y,z,[tmin,tmax])

指定 t 的取值范围为：tmin<t<tmax，在图形窗口中绘制符号方程 $x=x(t)$，$y=y(t)$，$z=z(t)$ 的图形。

例如，在 MATLAB 的命令窗口中输入如下命令，将得到如图 4-13 所示的图形.

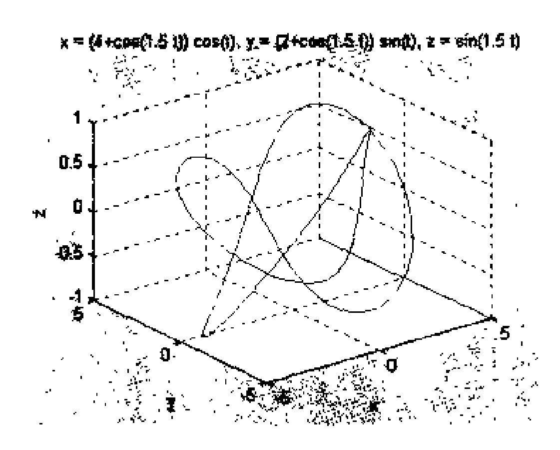


图 4-13　指定绘图区域的图形

```
x='(4+cos(1.5*t))*cos(t) ';
y='(2+cos(1.5*t))*sin(t) ';
z='sin(1.5*t) ';
ezplot3(x,y,z,[0,4*pi])
```

- ezplot3(x,y,z, 'animate')或者

ezplot3(x,y,z,[tmin,tmax], 'animate')

此外，对于比较复杂的曲线，ezplot3 命令还提供了一个选项'animate'，用于产生动态



的痕迹。

例如，下面的例子中，将产生一个跟踪曲线的小球。在 MATLAB 的命令窗口中输入如下代码，将得到如图 4-14 所示的图形。

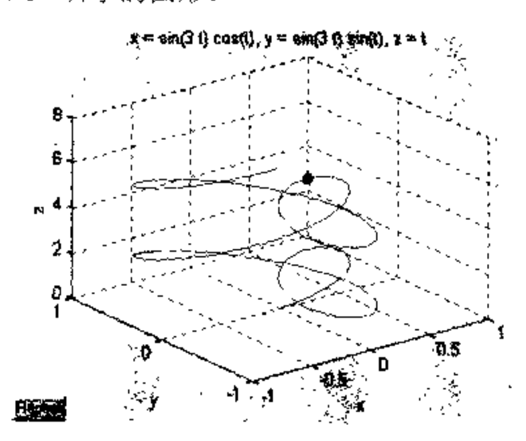


图 4-14　产生动态跟踪的曲线

```
x='sin(3*t)*cos(t)';
y='sin(3*t)*sin(t)';
z='t';
ezplot3(x,y,z,'animate')
```

单击【repeat】按钮，将演示曲线的产生过程。

2．ezmesh()函数的使用

- ezmesh(f)

对于符号函数 f(*x*,*y*)，ezmesh(*f*)在默认的绘图范围：

-2*pi<*x*<2*pi，-2*pi<*y*<2*pi

在图形窗口中绘制 *f*(*x*,*y*)的图形。

例如，下面的代码将在图形窗口中绘制 *f*(*x*,*y*)的曲线，结果如图 4-15 所示。



图 4-15　使用 ezmesh(*f*)绘制的图形

```
syms x y;
f='x*exp(-x^2-y^2)';
```



ezmesh(f)

- ezmesh(f,[xmin,xmax,ymin,ymax])或者 ezmesh(f,[a,b])

对于符号函数 f(x,y)，ezmesh(f) 在指定的绘图范围：

xmin<x<xmax，ymin<y<ymax

或者 a<x<b，a<y<b

在图形窗口中绘制 f(x,y)的图形。

- ezmesh(x,y,z)、ezmesh(x,y,z,[a,b])或者 ezmesh(x,y,z,[smin,smax,tmin,tmax])

这个命令将在图形窗口中绘制符号函数 $x=x(s,t)$，$y=y(s,t)$，$z=z(s,t)$ 的图形。

绘图范围为：-2*pi<s<2*pi，-2*pi<t<2*pi 或者 smin<s<smax，tmin<t<tmax。

例如，下面的代码将在图形窗口中绘制如图 4-16 所示的曲线。

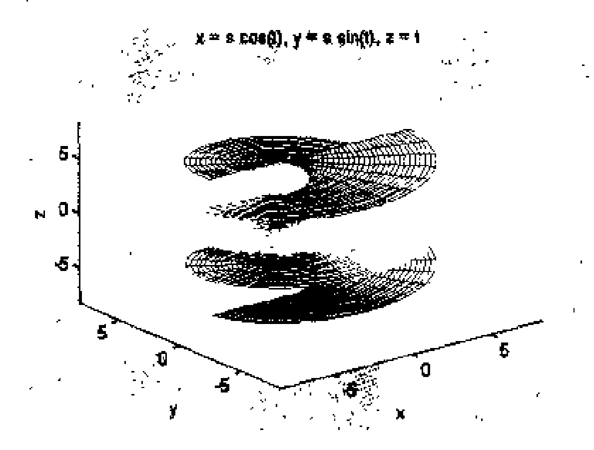


图 4-16　ezmesh(x,y,z)绘制的曲线

```
syms x y z s t
x='s*cos(t)';
y='s*sin(t)';
z='t';
ezmesh(x,y,z)
```

或者使用 ezmesh('s*cos(t)','s*sin(t)','t')

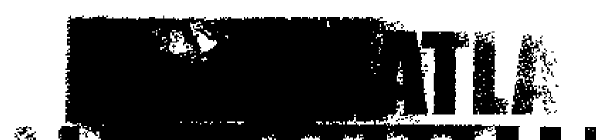
例如，下面的两个例子将在图形窗口中绘制如图 4-17 和图 4-18 所示的曲线。

例 1：

```
syms x y z s t
x='exp(-s)*cos(t)';
y='exp(-s)*sin(t)';
z='t';
ezmesh(x,y,z,[0,8,0,4*pi])
```

例 2：

```
syms x y z s t
x='(s-sin(s))*cos(t)';
y='(1-cos(s))*sin(t)';
z='s';
```




```
ezmesh(x,y,z,[-2*pi,2*pi])
```

对于绘制出来的三维图形，在图形窗口中单击该图形后，还可以对图形按任意角度进行旋转。

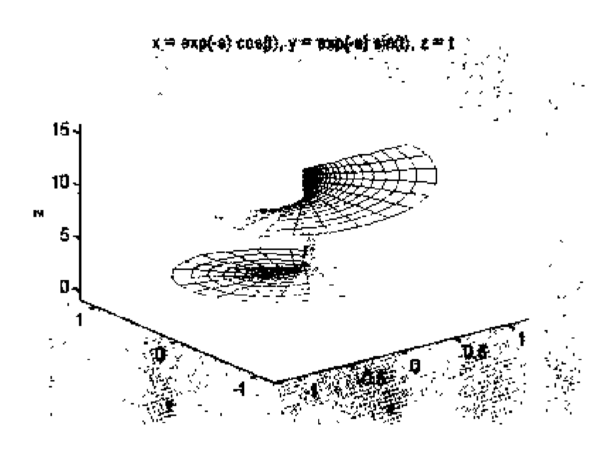


图 4-17　ezmesh(*x*,*y*,*z*,[smin,smax,tmin,tmax])的图形

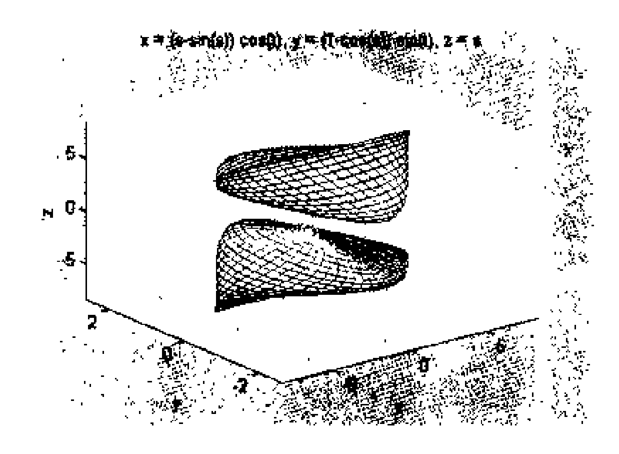


图 4-18　ezmesh(x,y,z,[a,b])绘制的图形

第 5 章　MATLAB 的可视化功能





MATLAB 语言提供了强大的图形绘制功能。在大多数情况下，用户只需要指定绘图方式，提供绘图数据，利用 MATLAB 提供的丰富的二维、三维图形函数，就可以绘制出所需的图形。

MATLAB 还对绘出的图形提供了各种修饰方法，使图形更为美观、精确。MATLAB 的图形系统是建立在诸如线、面等图形对象集合的基础之上的，用户通过设置对象特征来控制图形。

5.1 绘制二维图形

5.1.1 绘制简单的二维曲线

MATLAB 中最常用的绘图函数为 plot，根据 plot 函数参数的不同，可以在平面上绘制不同的曲线。plot 函数是将各个数据点通过连折线的方式来绘制二维图形的，若对曲线细分的话，曲线可以看成是由直线连接而成的。plot 命令的格式有以下几种：

（1）plot(y)

当 y 为一向量时，以 y 的序号作为 X 轴，按向量 y 的值绘制曲线。

（2）plot(x,y)

x,y 均为向量时，以 x 向量作为 X 轴，向量 y 作为 Y 轴绘制曲线。

（3）plot(x,y1,'option',x,y2,'option',…)

以公共的 x 向量作为 X 轴，分别以向量 y1,y2…的数据绘制多条曲线。每条曲线的属性由相应的选项'option'来确定。option 选项可以是表示曲线颜色的字符、表示线型格式的符号和表示数据点的标记。各个选项有的可以连在一起使用。曲线颜色、线型格式和标记如表 5-1 所示。

表 5-1　曲线颜色与线型格式

符　号	颜　色	符　号	颜　色	符　号	线　型	符　号	标　记	符　号	标　记
'b'	蓝色	'c'	青色	'-'	实线	'v'	▽	'x'	叉号
'g'	绿色	'k'	黑色	'--'	虚线	'^'	△	'+'	加号
'm'	洋红色	'r'	红色	':'	点线	'>'	▷	'pentagram'	五角星
'w'	白色	'y'	黄色	'-.'	点划线	'<'	◁	'diamond'	菱形
可以用一个 1×3 向量任意指定				'none'	无线	'.'	圆圈	'hexagram'	六角星
[r,g,b] 红、绿、蓝三原色						'*'	星号	'square'	

（4）plot(x1,y1,'option',x2,y2,'option',…)

分别以向量 x1，x2,…作为 X 轴，以 y1，y2…的数据绘制多条曲线，每条曲线的属性由相应的选项'option'来确定。曲线颜色、线型格式和标记如表 5-1 所示。

例如，在 MATLAB 的命令窗口中键入如下命令，可画出如图 5-1 所示的图形。



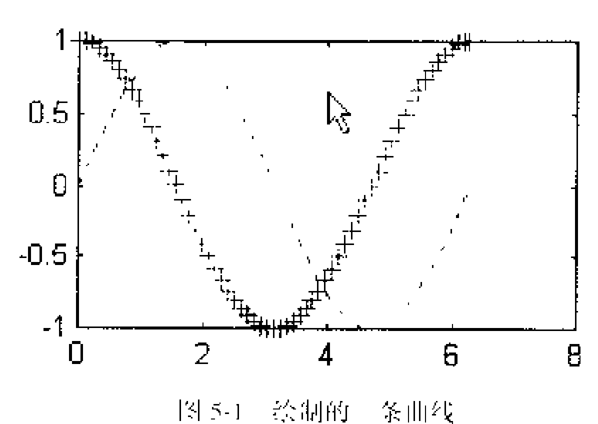


图 5-1 绘制的两条曲线

```
x=0:pi/30:2*pi;
y1=sin(x);
y2=sin(x+pi/2);
plot(x,y1,'r:',x,y2,'m+')
```

除 plot 命令外，MATLAB 还提供 line 命令，用于在图形窗口的任意位置画直线或折线。

line(x,y)

在当前图形窗口中绘制出一条由向量 x 和向量 y 的对应数据元素为数据点的折线。

例如，在 MATLAB 的命令窗口中输入如下命令，将得到如图 5-2 所示的图形。

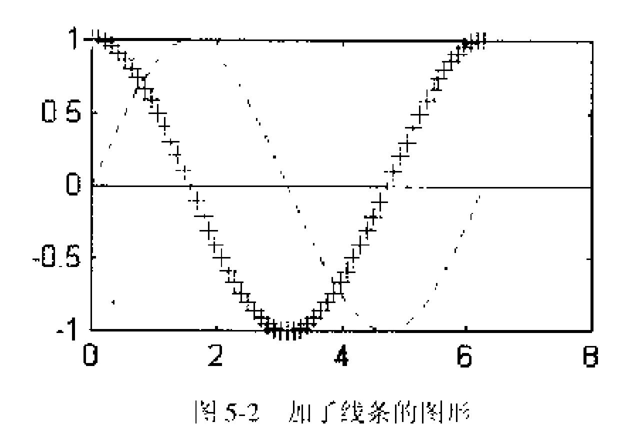


图 5-2 加了线条的图形

```
x=0:pi/30:2*pi;
y1=sin(x);
y2=sin(x+pi/2);
plot(x,y1,'r:',x,y2,'m+')
line([0 8],[0 0])
```

line 命令是 MATLAB 的低级绘图命令，它的功能由 plot 命令也可以实现。

5.1.2 离散序列图的绘制

当我们要处理离散量的时候，离散序列图就可以表示离散量的变化情况。MATLAB



用 stem 命令来实现离散序列图的绘制，stem 命令有如下几种格式：

（1）stem(y)

以 x=1，2，3…为各个数据点的 x 坐标，以 y 向量的各个对应元素为 y 坐标，在（x，y）坐标点画一个空心小圆圈，并连接一条线段到 X 轴。

（2）stem(x,y,'在 option')

以 x 向量的各个元素为 x 坐标，以 y 向量的各个对应元素为 y 坐标，在（x，y）坐标点画一个空心小圆圈，并连接一条线段到 X 轴。option 选项表示绘图时的线型、颜色，其取值见表 5-1。

（3）stem(x,y, 'filled')

以 x 向量的各个元素为 x 坐标，以 y 向量的各个对应元素为 y 坐标，在（x，y）坐标点画一个实心小圆圈，并连接一条线段到 X 轴。

例如，对于 $y = \sin x^2 e^{-x}$，下面的命令将产生如图 5-3 所示的图形。

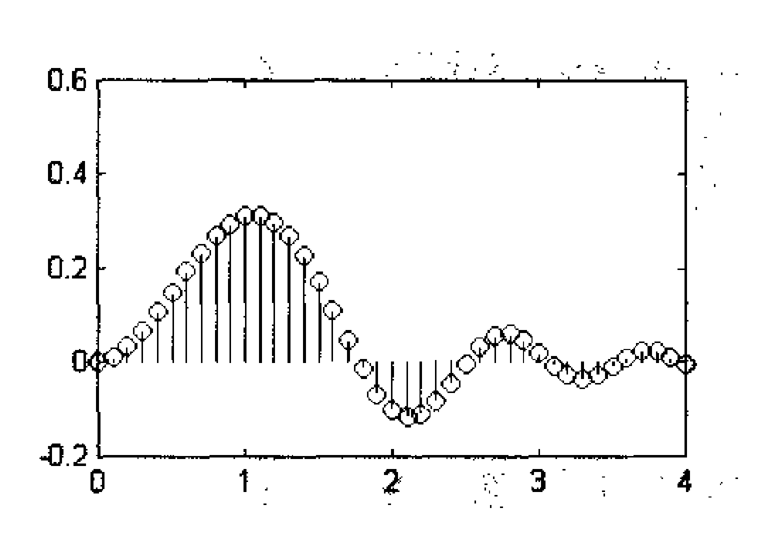


图 5-3　绘制的离散序列图

```
% Stem plot
x = 0:0.1:4;
y = sin(x.^2).*exp(-x);
stem(x,y)
```

5.1.3　二维图形的修饰

在利用 plot 函数绘图时，MATLAB 按默认设置及用户指定的数据绘制图形。MATLAB 还提供了一些图形函数，专门用于对由 plot 命令所画出的图形进行修饰，如坐标轴范围的设定（axis 命令）、加坐标轴名称（xlable，ylable 命令）、给图形加标题（title 命令）、加网格（grid 命令）、对图形进行文字标注（text 命令）等。



1．坐标轴的调整

MATLAB 可以自动地根据曲线数据的范围选择合适的坐标系，从而使得曲线尽可能清晰地显示出来，所以在一般情况下不必选择坐标系。但是，如果对 MATLAB 自动产生的坐标轴不满意，可以利用 axis 命令对坐标轴进行调整。axis 命令的调用格式为：

axis(xmin xmax ymin ymax)

这个命令将所画图形的 X 轴的大小范围限定在 xmin 和 xmax 之间，Y 轴的大小范围限定在 ymin 和 ymax 之间。

例如，在 MATLAB 窗口中键入如下命令，可画出如图 5-4（a）所示的图形。

```
x=0:1/100:2*pi;
y=sin(x);
plot(x,y)
line([0,2*pi],[0,0])
```

如果在最后一条命令后添加命令：

```
axis([0 2*pi -2 2])
```

可绘制出如图 5-4（b）所示的图形。其效果就好像对图 5-4（a）的图形进行了展宽。

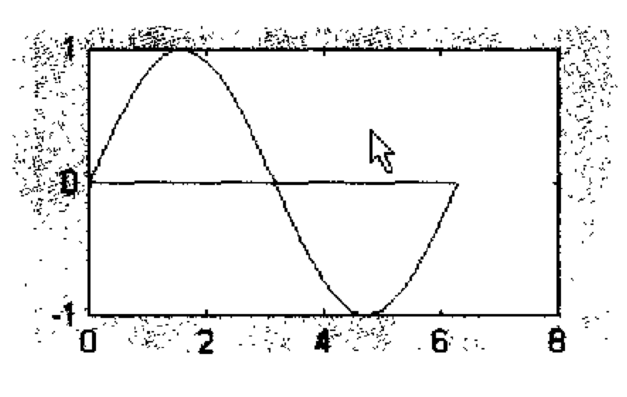


图 5-4（a）

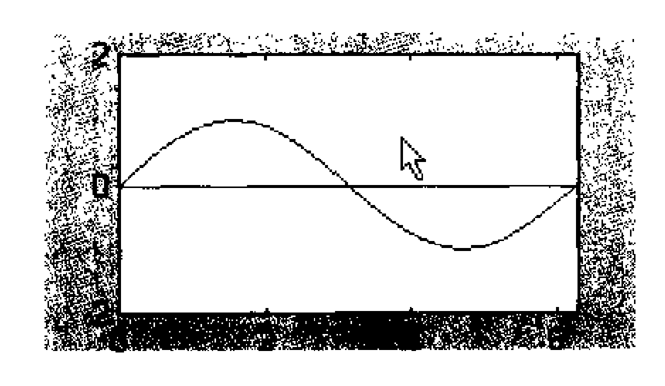


图 5-4（b）

2．标识坐标轴名称

通过 xlable（'string'）和 ylable（'string'）命令给 X 轴和 Y 轴加上标注。title（'string'）命令给图形加上标题。使用 grid on 或 grid off 命令在所画出的图形中添加或去掉网络线。

例如，在 MATLAB 命令窗口键入如下命令，得到如图 5-5 所示的图形。

```
x=0:1/100:2*pi;
y1=sin(x);
y2=cos(x);
plot(x,y1,'r-',x,y2,'b:')
grid on;
xlable('弧度值')
ylable('函数值')
title('正弦曲线和余弦曲线')
```

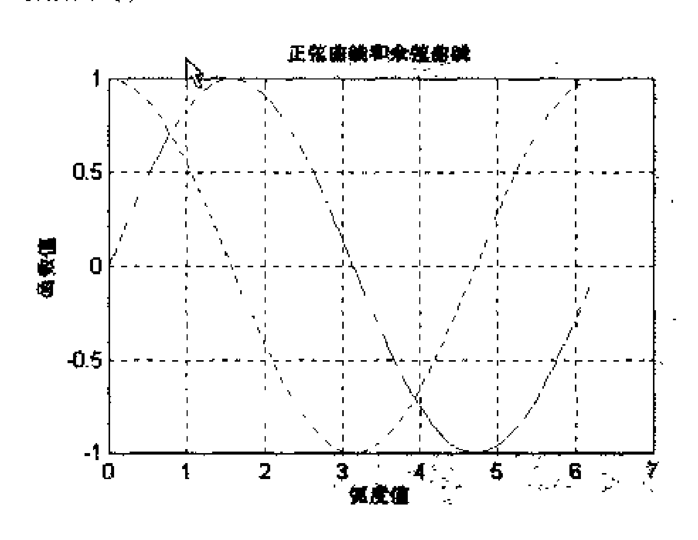


图 5-5　加上标识的图形

3. 在图形中加文本标注

对图 5-4 所示的图形，可以分别在两条曲线上加注文字。

MATLAB 允许用户在图形的任意位置加注一串文本。在任意位置加注文本有两种方式：使用坐标轴确定文字位置的 text 命令；使用鼠标确定文字位置的 gtext 命令。

- text(x,y,'string','option')

在图形的指定坐标位置(x,y)处，写出由 string 所给出的字符串。其中 x，y 的坐标的单位是由后面的 option 选项决定的。如果不加选项，则 x，y 的坐标单位和图中一致；如果选项为'sc'，表示坐标单位是取左下角为（0，0），右上角为（1，1）的相对坐标。

例如，在 MATLAB 的命令窗口键入如下命令，将得到如图 5-6 所示的结果。

```
x=0:1/100:2*pi;
y1=sin(x);
y2=cos(x);
plot(x,y1,'r-',x,y2,'b:')
grid on;
xlabel('弧度值')
ylabel('函数值')
```



```
title('正弦曲线和余弦曲线')
text(0.8,0.55,'正弦曲线','sc')
text(0.8,0.8,'余弦曲线','sc')
```

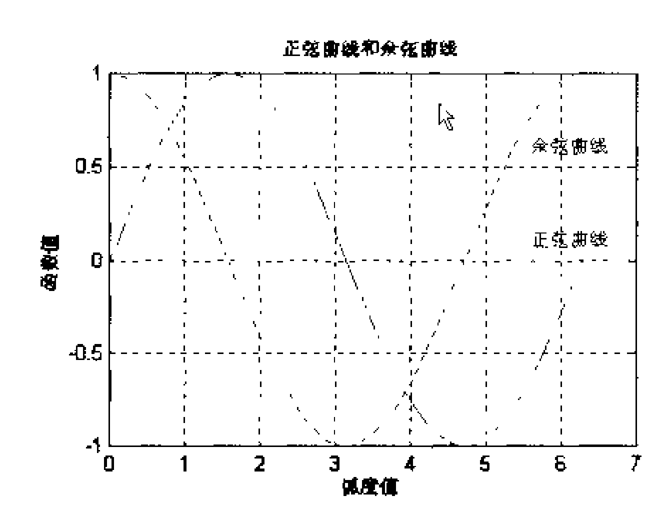


图 5-6　在指定位置加标注

- gtext('string')

利用鼠标，在图形的某一位置写出由 string 所给出的字符串。在 MATLAB 的命令窗口中输入 gtext 命令后，在图中将会出现一个十字形指针，用鼠标拖动到需要添加文字的地方，然后单击鼠标，就可以将 gtext 命令中的字符串添加到图形中。

例如，对于下面的代码，将在图形中使用鼠标添加标注，添加过程如图 5-7 所示。

```
x=0:1/100:2*pi;
y1=sin(x);
y2=cos(x);
plot(x,y1,'r-',x,y2,'b:')
xlabel('弧度值')
ylabel('函数值')
gtext('正弦曲线和余弦曲线')
```

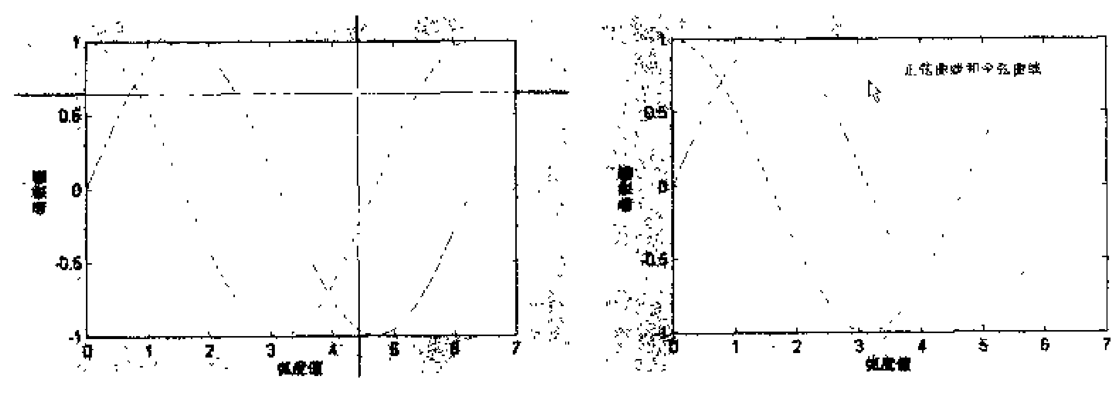


图 5-7　利用鼠标添加文本的过程



4．调整坐标轴状态

axis('string')

这个命令将坐标轴的状态调整为字符串'string'的指定的状态。各种常用字符串的含义如表 5-2 所示。

表 5-2　函数 axis('string')中字符串的常用功能

命令形式	命令功能
Axis([xmin xmax ymin ymax])	按照所给出的 X 轴和 Y 轴的最大、最小值选择坐标系
Axis('auto')	按照 x 和 y 和最大、最小值设定坐标系
Axis('square')	将当前图形设置为正方形图形
Axis('equal')	将 X，Y 坐标轴的单位设置为相等
Axis('normal')	关闭 axis equal 和 axis square 命令的作用
Axis('on')	打开网格线、XY 坐标轴用 label 命令添加的注释
Axis('off')	关闭网格线、XY 坐标轴用 label 命令添加的注释，保留用 text 和 gtext 命令添加的文本

例如，下面的程序将绘制一个单位圆。

```
t=0:1/100:2*pi;
x=sin(t);
y=cos(t);
plot(x,y);
grid on
```

结果如图 5-8 所示。由于计算机屏幕上 X 方向和 Y 方向的单位长度不一致，看起来像椭圆，使用命令：

Axis('square'); 或 Axis('equal');

将得到如图 5-8 所示的图形。注意，Axis('square')的含义是将 X 坐标轴长度与 Y 坐标轴长度调整为正方形；而 Axis('equal')是将 X 坐标轴和 Y 坐标轴的单位刻度调整成一样长。使用 Axis('normal')又可以恢复原状。

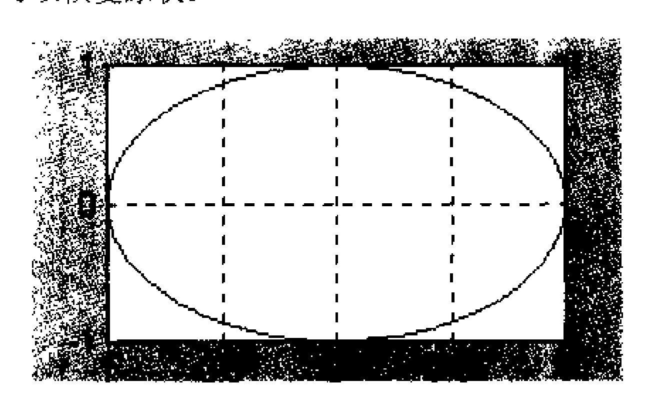


图 5-8（a）　进行刻度调整的圆



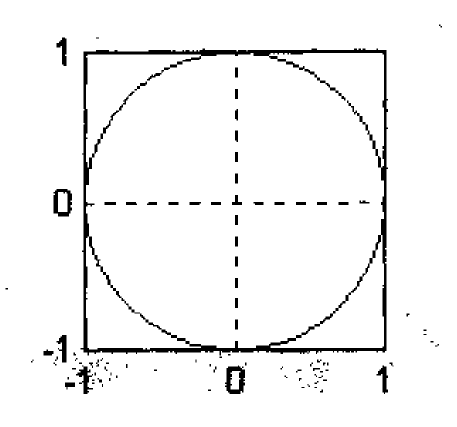


图 5-8（*b*）　未进行刻度调整的圆

5.2　绘制三维图形

MATLAB 提供了许多实现三维数据可视化的函数，可以在三维空间中绘制曲线或曲面。

5.2.1　三维折线及曲线的基本绘图命令

plot3(x1,y1,z1,'option1',x2,y2,z2,'option2',...)

plot3 是 MATLAB 绘制三维折线或曲线的基本命令。以 x1，y1，z1 所给出的数据分别为 *x*，*y*，*z* 坐标值，option1 为选项参数，以逐点连折线的方式绘制一个三维折线图；以 x2，y2，z2 所给出的数据分别为 *x*，*y*，*z* 坐标值，option2 为选项参数，以逐点连折线的方式绘制另一个三维折线图。

plot3 命令是以逐点连线的方式绘制三维折线，当各个数据点的间距较小时，绘制的就是三维曲线。plot3 命令的参数的含义与 plot 命令相类似，plot3 命令多了一个 *Z* 方向的参数，option 选项指定了线条的线型、颜色和数据点的表示记号，其取值见表 5-1。与二维曲线类似，在实际应用中，可根据各个数据的取值情况，取简单的格式。

例如，下面的代码将绘制出如图 5-9 所示的三维螺旋线。

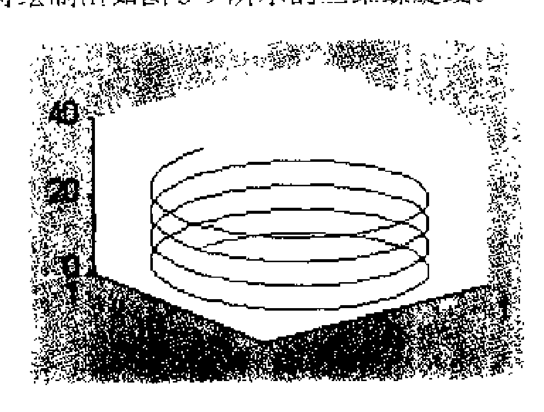


图 5-9　绘制的三维螺旋线





```
t=0:pi/50:8*pi;
x=sin(t);
y=cos(t);
plot3(x,y,t);
```

与二维图形类似，对于绘制的三维图形，也可以通过命令：

```
axis(xmin xmax ymin ymax zmin zmax)
```

调整三维图形各个坐标轴的显示范围，只不过这个命令比二维图形坐标轴调整多了一个 Z 轴方向。

同样地，可以对三维图形通过 title('string')命令加上标题，通过 zlabel('string')给 Z 坐标轴加上标注。

5.2.2 三维网格曲面的绘制

三维网格曲面图所构成的网格状表面由 ***X-Y*** 平面上的矩形栅格及相应的 Z 坐标构成，相邻点之间用直线连接。对于显示大型数据矩阵或双变量的函数是很有用的。

1．栅格数据点的产生

在绘制网格曲面之前，要知道各个四边形顶点的三维坐标值。绘制曲面的一般情况是，先知道二维坐标（x、y），然后利用函数公式计算 z 的坐标。二维坐标（x、y）是一种栅格形的数据点，可由 meshgrid 命令产生。

meshgrid(x,y)

由 x 向量和 y 向量通过复制的方法产生绘制图形时所需的栅格数依据 ***X*** 矩阵和 ***Y*** 矩阵。该命令产生栅格数据的方法是：将向量 x 作为矩阵 ***X*** 的一个行向量，并将向量 x 复制 length(y)次，以构成栅格数据点 ***X*** 矩阵；同样，将向量 y 作为矩阵 ***Y*** 的一个列向量，并将向量 y 复制 length(x)次，以构成栅格数据点 ***Y*** 矩阵。当 x 向量和 y 向量取值相同时，可省略一个参数。

例如，下面的示例将产生 ***X*** 和 ***Y*** 矩阵。

```
x=[1 2 3 4 5];
y=[7 8 9];
[X,Y]=meshgrid(x,y)
X=
   1  2  3  4  5
   1  2  3  4  5
   1  2  3  4  5
Y=
   7  7  7  7  7
   8  8  8  8  8
   9  9  9  9  9
```

例如，对于函数 $f(x,y)=xe^{-x^2-y^2}$，使用 meshgrid()命令生成绘图数据，用 plot3 命



令绘制图形，结果如图 5-10 所示。

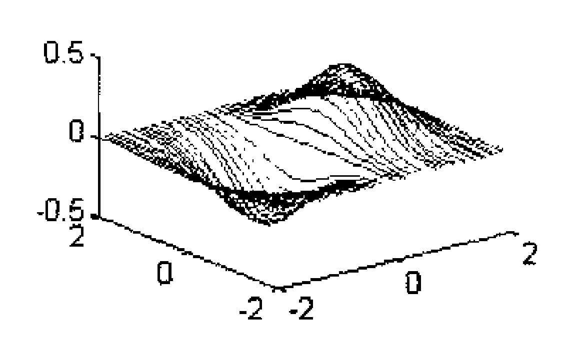


图 5-10　利用栅格数据绘制的图形

```
[X,Y] = meshgrid([-2:0.1:2]);
Z = X.*exp(-X.^2-Y.^2);
plot3(X,Y,Z)
grid on
```

2. **绘制三维网格曲面**

- mesh(X,Y,Z)命令绘制网格曲面

参数 **X**，**Y**，**Z** 都是矩阵，**X** 矩阵的行向量相同，**Y** 矩阵的列向量相同

- mesh(x,y,Z)命令绘制网格曲面

参数 x 和 y 分别是长度为 n 和 m 的向量，**Z** 是 $m\times n$ 矩阵。

- mesh(Z)命令绘制网格曲面

若 **Z** 是 $m\times n$ 矩阵，则栅格数据点的取法是：x=1:n，y=1:m。

例如，对于函数 $f(x,y)=\dfrac{\sin(\sqrt{x^2+y^2})}{\sqrt{x^2+y^2}}$，下面示例命令可绘制出如图 5-11（a）所示的网格曲面图形

```
x=-8:0.5:8;
y=x;
[X,Y]=meshgrid(x,y);
R=sqrt(X.^2+Y.^2)+eps;
Z=sin(R)./R;
mesh(X,Y,Z)
grid on
axis([-10 10 -10 10 -1 1])
```

显示或不显示网格曲面的隐藏线将对图形的显示效果有一定影响，MATLAB 提供了相应的控制命令 hidden。使用 hidden on 命令去掉网格曲面的隐藏线，hidden off 命令显示网格曲面的隐藏线。

如图 5-11 所示，图 5-11（a）的图形无隐藏线，右边的图形 5-11（b）显示了隐藏线。

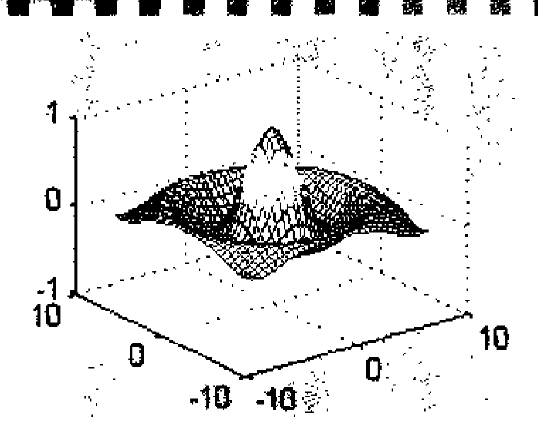


图 5-11（a）无隐藏线的曲面

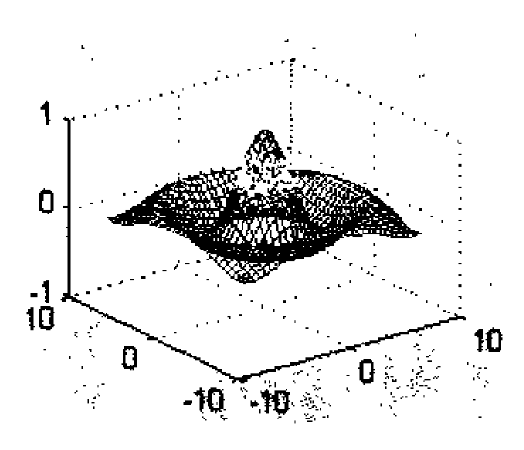


图 5-11（b）显示隐藏线的曲面

5.2.3　三维阴影曲面的绘制

MATLAB 提供了 surf 命令，用于绘制带阴影效果的三维曲面。surf 命令的完整格式是：

surf(X,Y,Z,C)

surf 命令与 mesh 命令类似，它的使用方法和参数的含义与 mesh 命令相同。在 surf 命令中，各个四边形表的颜色分布方式可由 shading 命令来指定。

shading facetd：表示截面颜色分布方式；

shading interp：表示插补式颜色分布方式；

shading flat：表示平面式颜色分布方式。

例如，对于函数：$f(x,y)=\dfrac{2\sin(\sqrt{x^2+y^2})}{\sqrt{x^2+y^2}}$

在分别设置了三种颜色分布方式后，显示的效果如图 5-12 所示。

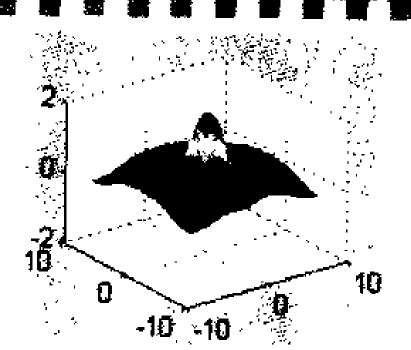


图 5-12（*a*）　shading faceted

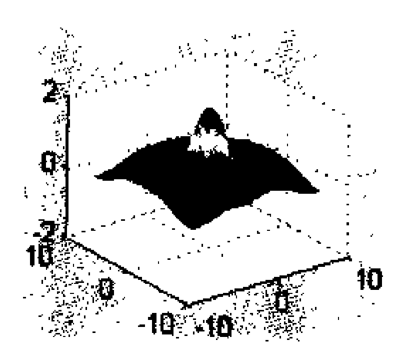


图 5-12（*b*）　shading interp

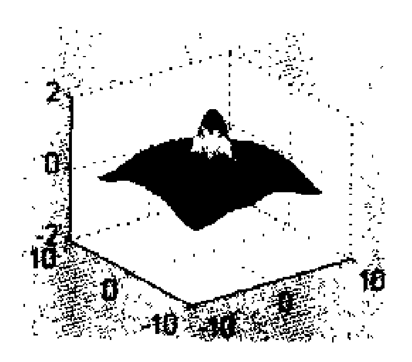


图 5-12（*c*）　shading flat

```
x=-8:0.5:8;
y=x;
[X,Y]=meshgrid(x,y);
R=sqrt(X.^2+Y.^2)+eps;
Z=2*sin(R)./R;
surf(X,Y,Z)
grid on
shading faceted
shading interp
shading flat
```

5.2.4 三维图形的视角变换

在 MATLAB 的三维图形绘制中，是按照如图 5-13 所示的方式来定义视角的，其中引入了两个角度方位角和仰角。

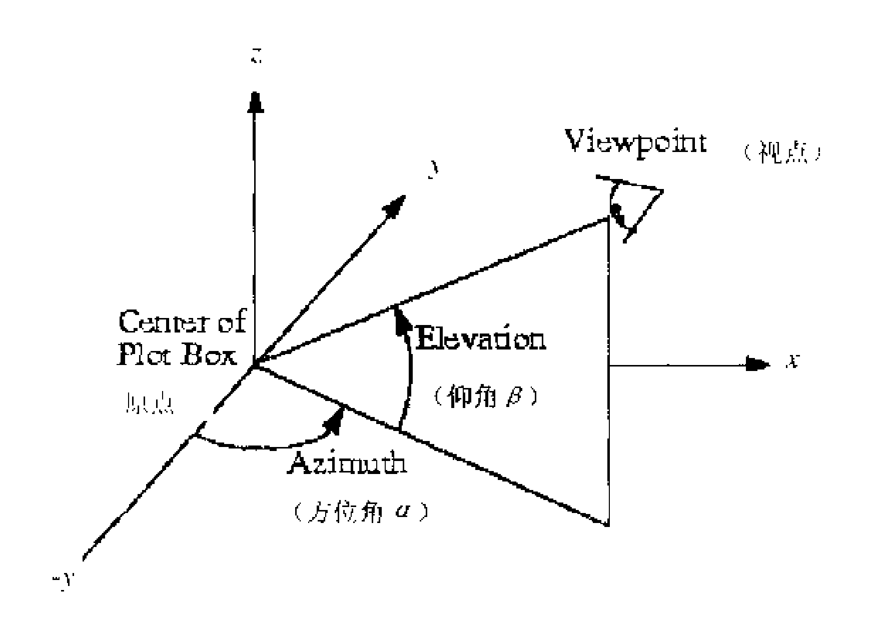


图 5-13 定义视角

方位角 α：视点在 x-y 平面上的投影与 Y 坐标轴负方向的夹角；

仰角 β：视点和 x-y 平面的夹角。

用户可以任意设定这两个角度以改变三维图形的视角。MATLAB 默认的视角定义为 α=-37.5°，β=30°。

MATLAB 提供了 view()函数来改变和的值，view()函数的调用格式为：

View(AZ,EL)、view([AZ,EL])或 view(x,y,z)

其中 AZ 和 EL 分别是方位角和仰角。view(x,y,z)指定视点的位置是(x,y,z)。

view(2) 设定[AZ，EL]=[0，90]，即观测 x-y 平面的显示效果。view(3) 设定[AZ，EL]=[-37.5，30]，即默认视点位置。

例如，对于函数 $f(x,y)=xe^{-x^2-y^2}$，在 MATLAB 的命令窗口输入如下代码：

```
[X,Y] = meshgrid([-2:.25:2]);
Z = X.*exp(-X.^2 -Y.^2);
surf(X,Y,Z);
```

这时，将得到如图 5-14（a）所示的图形。当我们变换视点后，例如使用命令：

```
view([180 0]);
```

将得到如图 5-14（b）所示的图形。同样地，图 5-14（c）和 5.14（d）是使用命令：view([0 0]); 和 view([0 90]) 的结果。



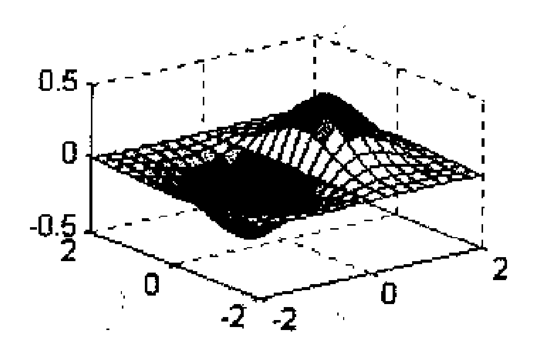


图 5-14（*a*）　view([-37.5 30])

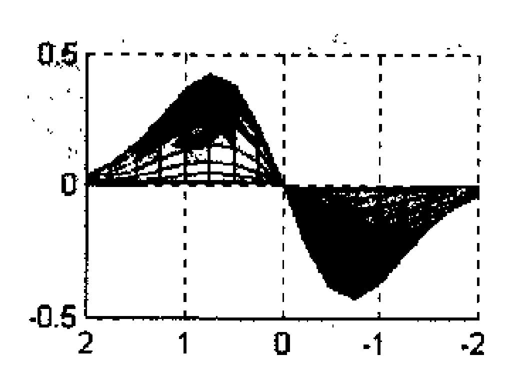


图 5-14（*b*）　view(180 0)

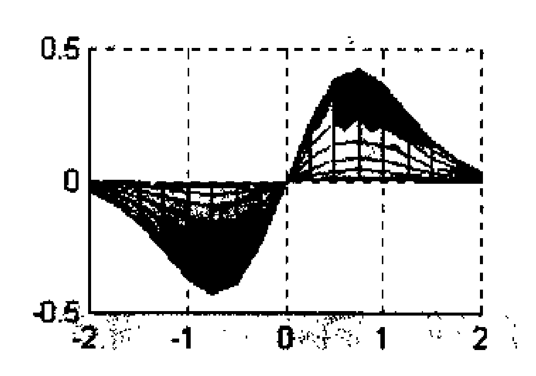


图 5-14（*c*）　view(0 0)

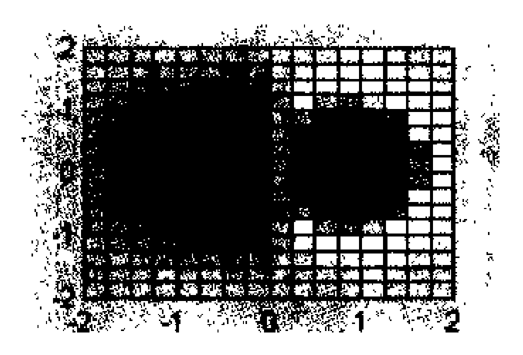


图 5-14（*d*）　view(0 90)

5.3 图形窗口的控制与表现

5.3.1 创建或打开图形窗口

通常，在 MATLAB 的命令窗口中运行一个绘图指令后，就自动创建一个名为“Figure No.1”的图形窗口，此后，这个窗口被当做当前窗口，之后绘图指令所画的图形都将出现在这个“Figure No.1”图形窗口，或者把原来的图形覆盖掉，或者叠加在原来的图形上。

如果想保留原来图形，那么可以使用命令来对图形窗口进行操作。

- figure 命令 每调用一次就打开一个新的图形窗口。
- figure（*n*）命令 创建或打开第 *n* 个图形窗口，使之成为当前窗口。

另外，可以使用 clf 命令，清除当前图形窗口中的所有内容，使用 cla 命令清除当前图形窗口的图形，保留其坐标。

5.3.2 图形重叠

在 MATLAB 中可以在同一坐标系中画出多幅图形。前面介绍的 plot 命令，就能在同一坐标系中画出多幅图形。但 plot 命令在执行时首先将当前图形窗口清屏，我们看到的是最后一条 plot 命令绘制的图形。MATLAB 提供了 hold 命令，将当前图形窗口的图形保留，利用多条 plot 命令绘制多幅图形。

hold on 命令：保留当前窗口的图形；hold off 命令：解除 hold on 命令。

例如，下面的代码在同一坐标系中绘制 3 幅图形，如图 5-15 所示。

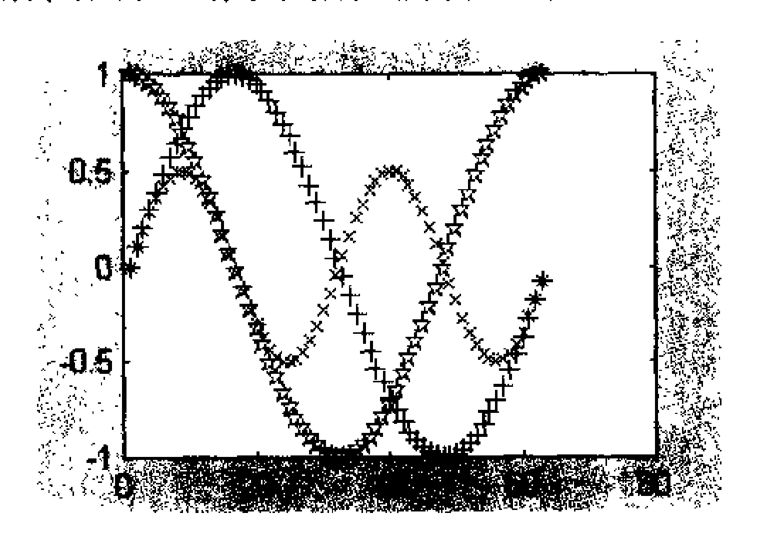


图 5-15 保留图形在同一窗口中

```
x=0:1/10:2*pi;
y1=sin(x);
y2=cos(x);
```



```
y3=sin(x).*cos(x)
plot(y1,'b+')
hold on
plot(y2,'mpentagram');
plot(y3,'kx')
hold off
```

5.3.3 图形窗口分割

利用图形窗口分割函数 subplot（），也可以在同一个图形窗口中绘制多幅图形。

subplot(m,n,p)

将当前绘图窗口分割成 *m* 行、*n* 列，并且现在在其中的第 p 个区域绘图。

各个绘图区域以“从左到右，先上后下”的原则来编号。MATLAB 允许每个绘图区域以不同的坐标系单独绘制图形。在使用了 subplot 命令分割窗口之后，可以用 subplot(1,1) 或 clf 命令回到一个窗口的状态。

例如，在 MATLAB 的命令窗口中输入如下代码，将图形窗口分割成四个区域，并分别绘制图形，如图 5-16 所示。

```
t = 0:pi/20:2*pi;
[x,y] = meshgrid(t);
subplot(2,2,1)
plot(sin(t),cos(t))
axis equal
subplot(2,2,2)
z = sin(x)+cos(y);
plot(t,z)
axis([0 2*pi -2 2])
subplot(2,2,3)
z = sin(x).*cos(y);
plot(t,z)
axis([0 2*pi -1 1])
subplot(2,2,4)
z = (sin(x).^2)-(cos(y).^2);
plot(t,z)
axis([0 2*pi -1 1])
```

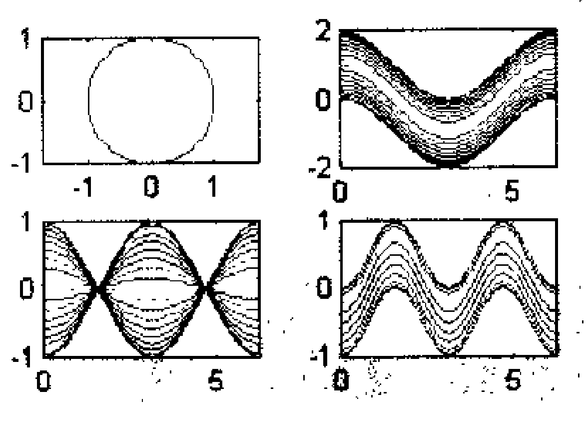


图 5-16 图形窗口的分割

例如，在 x-y 平面内选择一个区域，绘出下面这个二元函数的三维表面图形。

对于函数 z，定义如下：

$$z = 3(1-x)^2 e^{-x^2-(y+1)^2} - 10(\frac{x}{5} - x^3 - y^5)e^{-x^2-y^2} - \frac{1}{3}e^{-(x+1)^2-y^2}$$

在 MATLAB 的命令窗口中输入如下命令，将图形窗口分割成二个区域，并分别绘制 peaks 函数图形，显示结果如图 5-17 所示。

```
[X,Y]=meshgrid(-3:1/8:3);
z=3*(1-X).^2.*exp(-(X.^2)-(Y+1).^2)...
-10*(X/5-X.^3-Y.^5).*exp(-X.^2-Y.^2)...
-1/3*exp(-(X+1).^2-Y.^2);
subplot(1,2,1)
mesh(X,Y,z)
subplot(1,2,2)
surf(X,Y,z)
shading interp
```

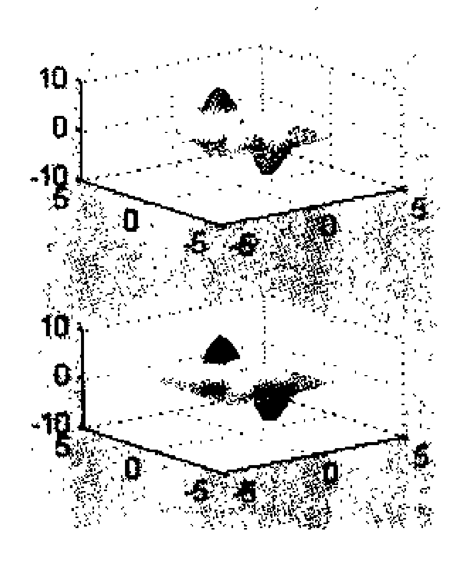


图 5-17 垂直分割图形窗口绘制的图形



5.4　图形对象及其属性设置

5.4.1　MATLAB 的图形对象

在 MATLAB 绘制出的图形中，各个图形元素是相互独立的，可单独进行修改处理。这种独立的图形元素称为图形对象。这些图形对象具有能控制其特征的属性，它们之间的关系如图 5-18 所示。

MATLAB 中的这些图形对象从根对象（root）开始，构成一种层次关系。在图 5-18 中，位于左边是的父对象，右边的是左边父对象的子对象。

当我们调用 plot 命令绘制二维曲线时，MATLAB 的执行过程大致如下：

（1）使用 figure 命令，在屏幕（root）对象上生成一个图形窗口（Figure 对象）。

（2）使用 axis 命令，在图形窗口内生成一个绘图区域（Axes 对象）。

（3）最后用 line 命令在 Axes 指定的区域内绘制线条（Line 对象）。

因此，MATLAB 所绘制的图形是由基本的图形对象组合而成的，可以改变图形对象的属性来设置所绘制的图形。

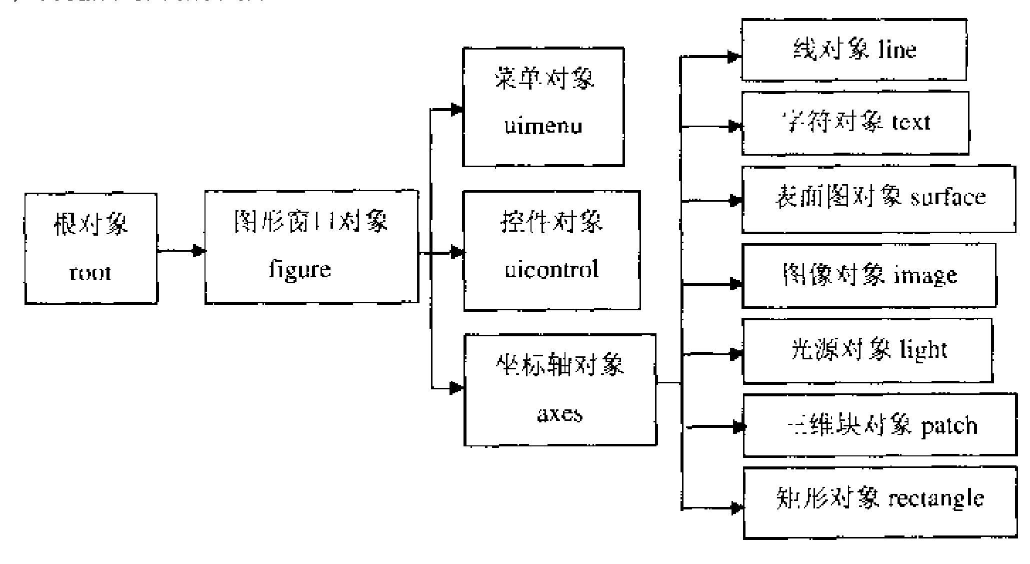


图 5-18　MATLAB 中的图形对象关系

5.4.2　句柄——图形对象的标识

所谓句柄，指的是某个图形对象的记号，MATLAB 给图形中的各个图形对象指定个句柄（handle），由句柄惟一地标识要操作的图形对象。

对于 root 对象，它的句柄是屏幕，这是 MATLAB 的规定，不用重新生成，root 对象的句柄值为 0。




Figure 对象的句柄生成指令是：

handle =figure('PropertyName',PropertyValue,...)，即可以在创建新窗口时直接设置其属性值。用户通过这样的方式打开图形窗口，并返回该窗口的句柄，以后就可以对其属性进行进一步的设置。

在 MATLAB 中，允许打开多个图形窗口，每个窗口有一个对应自己的句柄，因此，对于 Figure 对象，MATLAB 还提供了函数：

handle=gcf

其作用是返回当前窗口的句柄到 handle 变量。

Axes 对象是指在图形窗口中所设置的一个坐标轴，Axes 对象的句柄生成指令是：

handle =axes('PropertyName',PropertyValue,...)，即可以直接设置其属性值。

此外，利用 plot、plot3、mesh、surf 等函数绘图时，在这些命令中都包括有自动生成 Axes 对象的命令。由于 Axes 对象是一个经常要用到的图形对象，MATLAB 还提供函数：

handle=gca

其作用是返回当前坐标轴的句柄到 handle 变量。

Text 对象是指图形窗口中的一串文字，Text 对象可由 text 指令生成，另外，xlabel、ylabel、zlabel、title 等设置字符串的命令都包括有自动生成 Text 对象的命令。

5.4.3 图形对象属性的获取与设定

MATLAB 为不同的图形对象提供了很多控制其特征的属性。如 figure 对象的 color 属性可控制图形窗口的背景颜色，axes 对象的 Xlabel 属性设置 *X* 轴坐标的名称，Xgrid 属性设置是否在 *X* 轴的每一个刻度线画格线等。

不同的图形对象有不同的属性，可以通过 get 和 set 命令来获取或设置其属性值。

1．获取属性

PropertyValue=get(handle, 'ProperName')

获取指定图形对象的某个指定属性的属性值。对于句柄为 handle 的图形对象，返回其由 PropertyName 指定的属性的当前值，并将属性保存到变量 PropertyValue 中。

可以用 get(handle) 来获取某一对象的所有属性值。例如，使用 get(gcf)获取当前图形窗口的全部属性的当前属性值。

如果返回某一对象的所有属性的默认值，可使用 get(handle,'Default') 来获取。如果要返回某一属性的默认值，则在其属性名前加'Default'，此时的命令格式为：

PropertyValue=get(handle,'DefaulObjectTypeProperName')

2．属性的设置

set(handle,'ProtertyName1',ProtertyValue1,'PropertyName2',PropertyValue2,...)

该语句设定指定图形对象的某些指定属性的属性值。对于句柄为 handle 的图形对象，将 ProtertyName1 属性的属性值设定为 ProtertyValue1，将 ProtertyName2 属性的属性值设定为 ProtertyValue2，...

当要显示某一对象所有可设定的属性名称及其可能的取值时，可使用 set(handle)命令：



当要显示某一对象的某一属性的可能的取值时，使用 set(handle, 'ProtertyName')命令。

例如，在 MATLAB 命令窗口中键入下列代码得到如图 5-19（*a*）所示的结果。

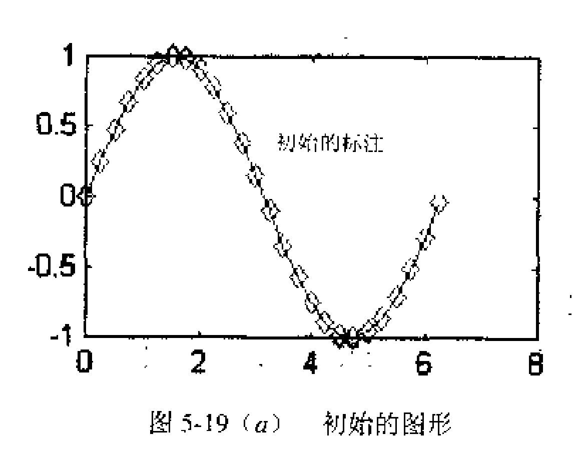


图 5-19（*a*）　初始的图形

```
x=0:1/4:2*pi;
y=sin(x);
hp=plot(x,y,'r-diamond')
ht=gtext('初始的标注')
```

这里返回的两个句柄：曲线句柄 hp 和字符句柄 ht，通过下面的语句修改曲线和标注，得到如图 5-19（*b*）所示的结果。

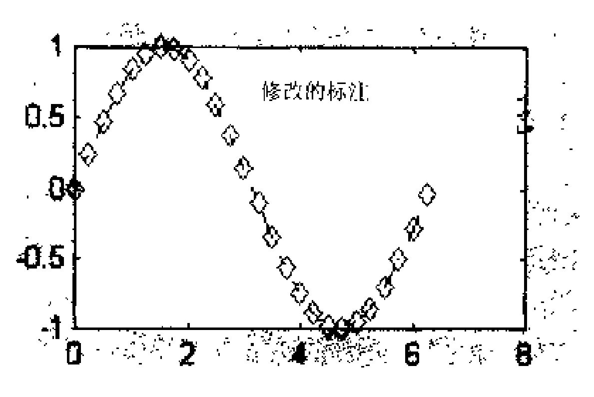


图 5-19（*b*）　改变属性后的图形

```
set(hp,'linestyle','-.','color','b');
set(ht,'string','修改的标注','FontSize',18,'Rotation',10);
```

在这两个命令中，首先改变的曲线的线型和颜色，然后更新字符串的内容和字号，并将其旋转了 10°。

5.4.4　图形对象常用属性

在 5.4.1 节中所给的 MATLAB 的图形对象具有很多属性。下面就一些常用对象的常用属性做一介绍。



1．坐标轴对象的常用属性

Box 属性：表示是否需要坐标轴上的方框，可以是'on'和'off'，默认值是'on'。

ColorOrder 属性：设置多条曲线的颜色顺序，设置值为 $n\times3$ 矩阵，也可以由 colormap()函数来设置。

GridlineStyle 属性：网格线类型，如实线、虚线等，其设置类似 plot()命令的选项，默认值为空。

NextPlot 属性：表示坐标轴图形的更新方式，默认值是'replace'，表示重新绘制图形，而且'add'选项表示在原来的图形上叠加，它相当于使用 hold on 命令的效果。

Title 属性：本坐标轴标题的句柄。具体内容由 title()函数设定，由此句柄可以访问原来的标题。

Xlabel 属性：X 轴标注的句柄，其内容由 xlabel()函数设定。类似的还有 Ylabel 属性和 Zlabel 属性等。

XDir 属性：X 轴的方向，可以选择'nomal'（正向）或'rev'（逆向）。类似的有 YDir 属性和 ZDir 属性等。

XGrid 属性：表示 x 轴是否加网格线，可以是'on'和'off'。类似的有 YGrid 属性和 ZGrid 属性等。

XLim 属性：表示 x 轴上下限，以向量[Xmin,Xmax]的形式给出，此外类似的还有 YLim 属性和 ZLim 属性等。前面介绍的 axis 函数实际上是对这些属性的直接赋值。

XTtick 属性和 XTickLabel 属性：XTtick 属性将给出 x 轴上标尺点值的向量，而 XTickLabel 属性将存放这些标尺点上的标记字符串。对 y 轴和 z 轴也有相应的标尺属性。

Color 属性：设置坐标轴对象的背景颜色，属性是一个 1×3 的颜色向量。默认是[1 1 1]，即白色。

FontAngle 属性：坐标轴标记文字的倾斜形式，如'normal'（正常）和'italic'（斜体）等。

FontName 属性：坐标轴标记文字的字体的名称。如'Times New Roman'和'Courier'等。

FontSize 属性：坐标轴标记文字的字号大小。默认是 10 磅。

FontWeight 属性：坐标轴标记文字的字体是否加黑。可以选择'light'、'normal'（默认）、'demi'和'bold'四个选项，颜色逐渐加深。

2．字符对象的常用属性

Color 属性：字符的颜色，属性是一个 1×3 的颜色向量。

FontAngle 属性：字体倾斜形式，如'normal'（正常）和'italic'（斜体）等。

FontName 属性：字体的名称。如'Times New Roman'和'Courier'等。

FontSize 属性：字号大小。默认以磅为单位。

FontWeight 属性：字体是否加黑。可以选择'light'、'normal'（默认）、'demi'和'bold'四个选项，颜色逐渐加深。

Rotation 属性：字体旋转的角度。

String 属性：构成字符对象的字符串。

Editing 属性：是否允许交互式修改。可以是'on'和'off'。

例如，在 MATLAB 的命令窗口中输入如下代码，将得到如图 5-20（a）所示的图形。



```
x = 0:0.1:4;
y = sin(x.^2).*exp(-x);
hl=plot(x,y);
```

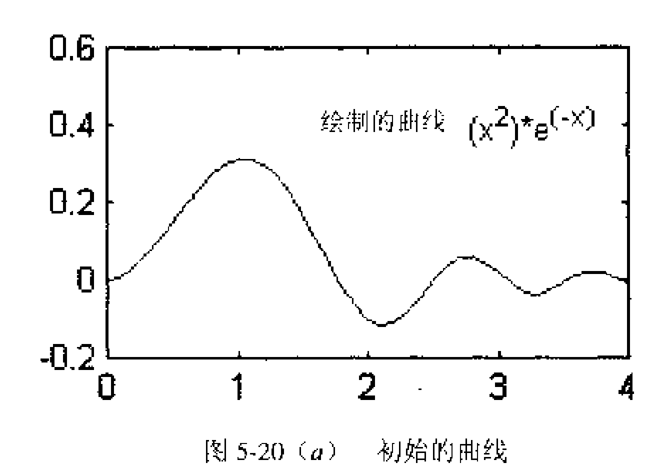


图 5-20（*a*）　初始的曲线

```
hc=text(1.18,0.4,'绘制的曲线 sin(x^2)*e^{(-x)}');
```

对于这个图形，通过 set()函数设置坐标轴对象和字符对象的属性，例如，输入如下代码，将得到如图 5-20（*b*）所示的图形。

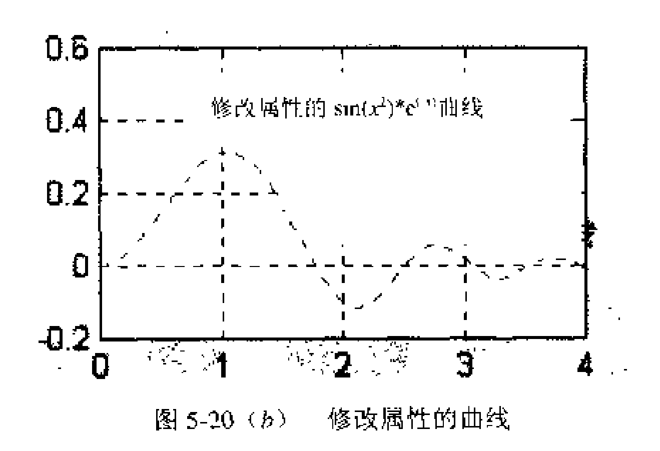


图 5-20（*b*）　修改属性的曲线

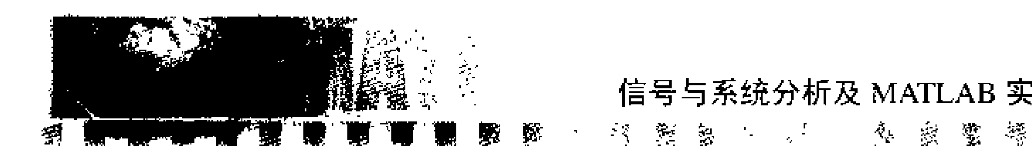
```
set(gca,'XGrid','on','YGrid','on');
set(hl,'linestyle',':');
set(hl,'Color','red');
set(hc, 'string', '修改属性的 sin(x^2)*e^{(-x)}曲线');
set(hc, 'fontsize',10, 'rotation',-18);
```

3. 曲线对象的常用属性

在调用 plot()语句时，将获得所绘制曲线的句柄，如果想改变曲线的颜色等参数，则调用 set()函数来改变其属性。

Color 属性：表示曲线的颜色，属性是一个 1×3 的颜色向量，其中三个元素分别表示三原色的值，范围在 0 到 1 之间。也可以用表 5-1 所示的颜色符号来表示。

LineStyle 属性：表示曲线的线型，如表 5-1 所示。



LineWeight 属性：表示曲线的线宽度默认值为0.5。

Maker 属性：表示曲线上的标号类型，如表5-1所示，若无标号，值为'none'。

MakerSize 属性：表示标号大小。其默认值为6。

XData 属性：该曲线对象的 x 轴数据。值为向量或矩阵，另外还定义了 YData 和 ZData 矩阵，可以通过曲线的句柄获取图形的相应数据。

例如，在 MATLAB 的命令窗口中输入如下代码，将绘制一条曲线。

通过设置其属性后，得到如图5-21所示的效果。

```
%propertiy
x=0:1/4:2*pi;
y=sin(x);
hndl=plot(y);
set(hndl,'Color','c');
set(hndl,'LineWidth',2);
set(hndl,'Marker','o','MarkerSize',9);
set(gca,'color','y');
```

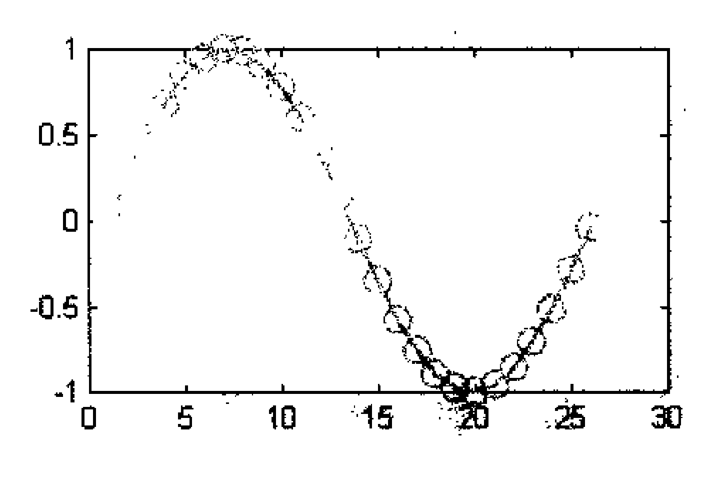


图5-21　曲线对象的属性设置

5.4.5　MATLAB 5.3 的图形可视编辑工具

除了可以利用 set()函数设置图形对象的属性外，MATLAB 5.3 版的图形窗口还提供了可视图形编辑工具。

MATLAB 5.3 版的图形窗口如图5-22所示，在这个窗口提供了一个工具栏，允许用户在图上标记字符、直线和箭头等。

在图形窗口的【File】菜单中，选择【Property Editor】菜单项则将打开属性编辑界面，双击 figure#1 标志，列出当前图形窗口下的全部子对象，其中一项为 axes，即坐标轴。双击坐标轴标志，将展开下一级子对象，选中一个子对象，就可以直接设置对象的属性。



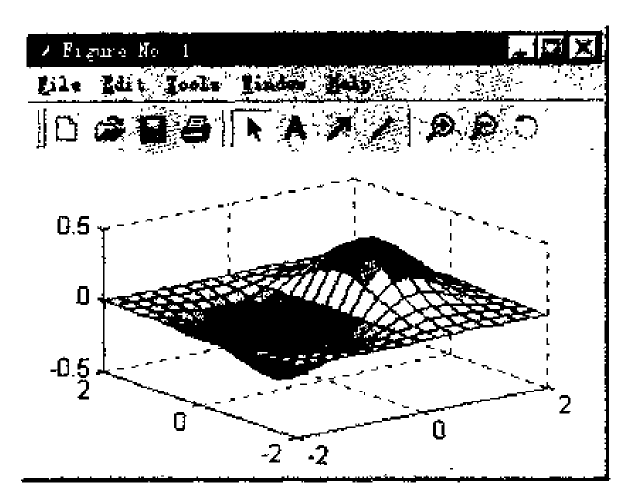


图 5-22 MATLAB 5.3 的图形窗口界面

在图形窗口中的【Tool】菜单下，当选中了一条直线或曲线时，可通过【Line Properties】选项设置曲线对象属性；当选中了一个字符标记时，可通过“Text Properties”选项设置选中字符的字体。同样地，选择【Tool】菜单下的【Axes Properties】选项，将打开坐标轴属性编辑对话框，供用户直观地设置坐标轴属性。

在工具栏中，提供了三维图形的视角变换功。单击工具栏上的旋转按钮，就可以进行视角变换，自由地旋转得到的三维图形。

另外，对于图形窗口的工具栏，单击编辑按钮（箭头）则可以进入编辑状态，用户可以选取图形窗口中的任意一个对象，此时该对象上会出现选中标志。我们可以很容易地移动选中的字符对象，还可以选择添加文字工具、下划线和箭头对图形进行编辑。

1. 坐标轴对象的属性设置

利用图形窗口工具栏上的编辑按钮（箭头），可以对选中的图形对象进行编辑

例如，在 MATLAB 的命令窗口中输入如下代码，将在图形窗口中绘制 $y=\sin(x)$的曲线。

```
x=0:1/10:2*pi;
y=sin(x);
plot(y)
```

单击图形窗口工具栏上的编辑按钮（箭头），则可以进入编辑状态，如图 5-23 所示。单击坐标轴对象，此时坐标轴对象上将出现选中标志。

双击坐标轴对象，或者在坐标轴对象上单击右键，在快捷菜单中选择【Properties】，将打开坐标轴属性设置窗口，如图 5-24 所示。

在对话窗口中可以设置坐标轴对象的标题、*X* 坐标轴的标注、*Y* 坐标轴的标注等等。

2. 曲线对象的属性设置

利用图形窗口工具栏上的编辑按钮（箭头），还可以对曲线对象进行编辑。

例如，对于上面的这个例子，在图形窗口中单击工具栏上的编辑按钮（箭头），进入编辑状态，单击曲线对象，此时曲线对象上将出现选中标志。

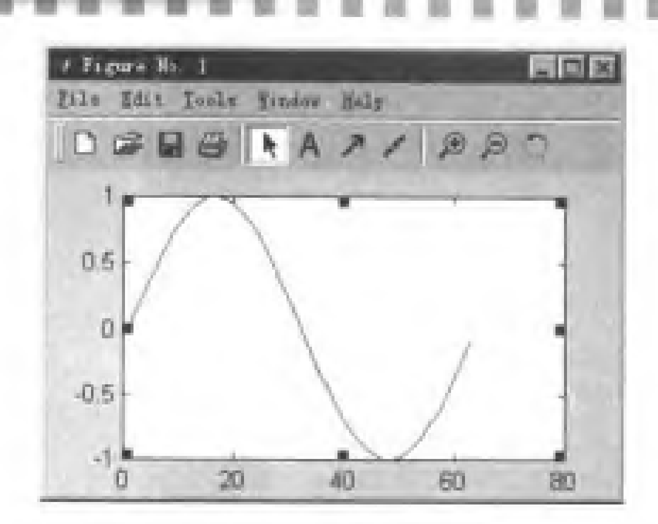


图 5-23　选中图形对象后可进行属性设置

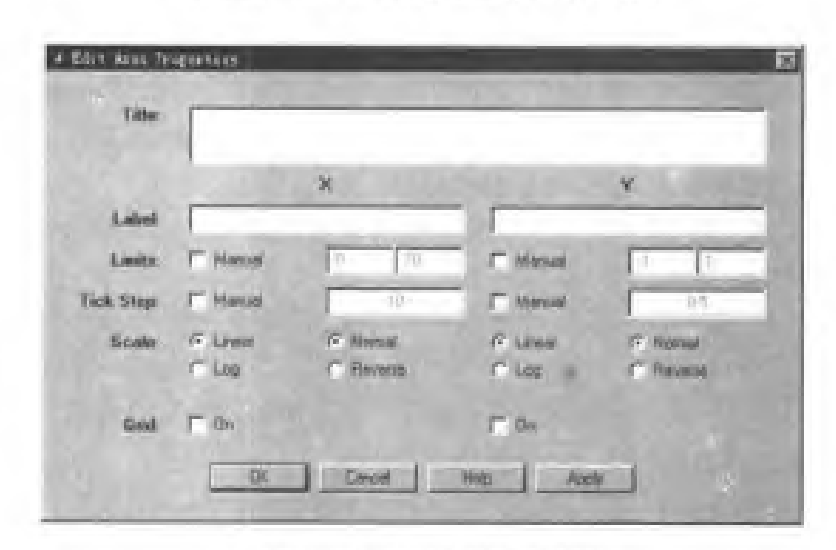


图 5-24　坐标轴对象属性设置窗口

双击曲线对象，或者在曲线对象上单击右键，在快捷菜单中选择【Properties】，将打开曲线属性设置窗口，如图 5-25 所示。

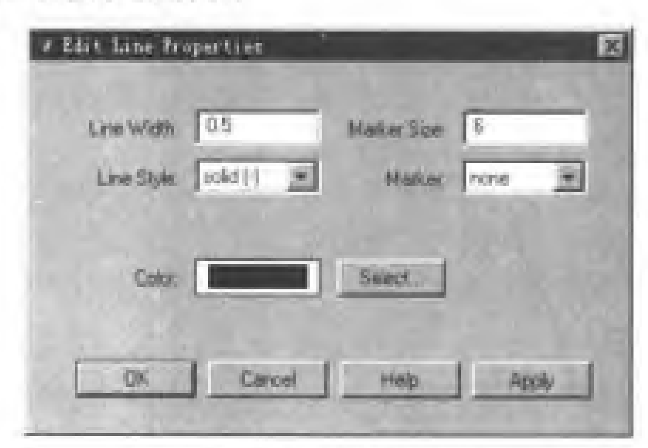


图 5-25　曲线对象属性设置



在这个对话窗口中可以设置曲线对象的线型、线条宽度、数据标志（Marker）、标志大小（Marker Size）、颜色等属性。

例如，我们设置曲线的数据标志（Marker）为五角星（hexagram），标志大小(Marker Size)为 8，单击【OK】按钮，返回图形窗口，再单击【编辑】按钮（箭头）后，得到如图 5-26 所示的结果。

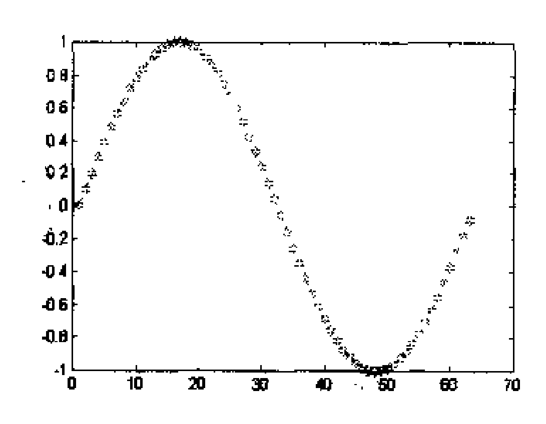
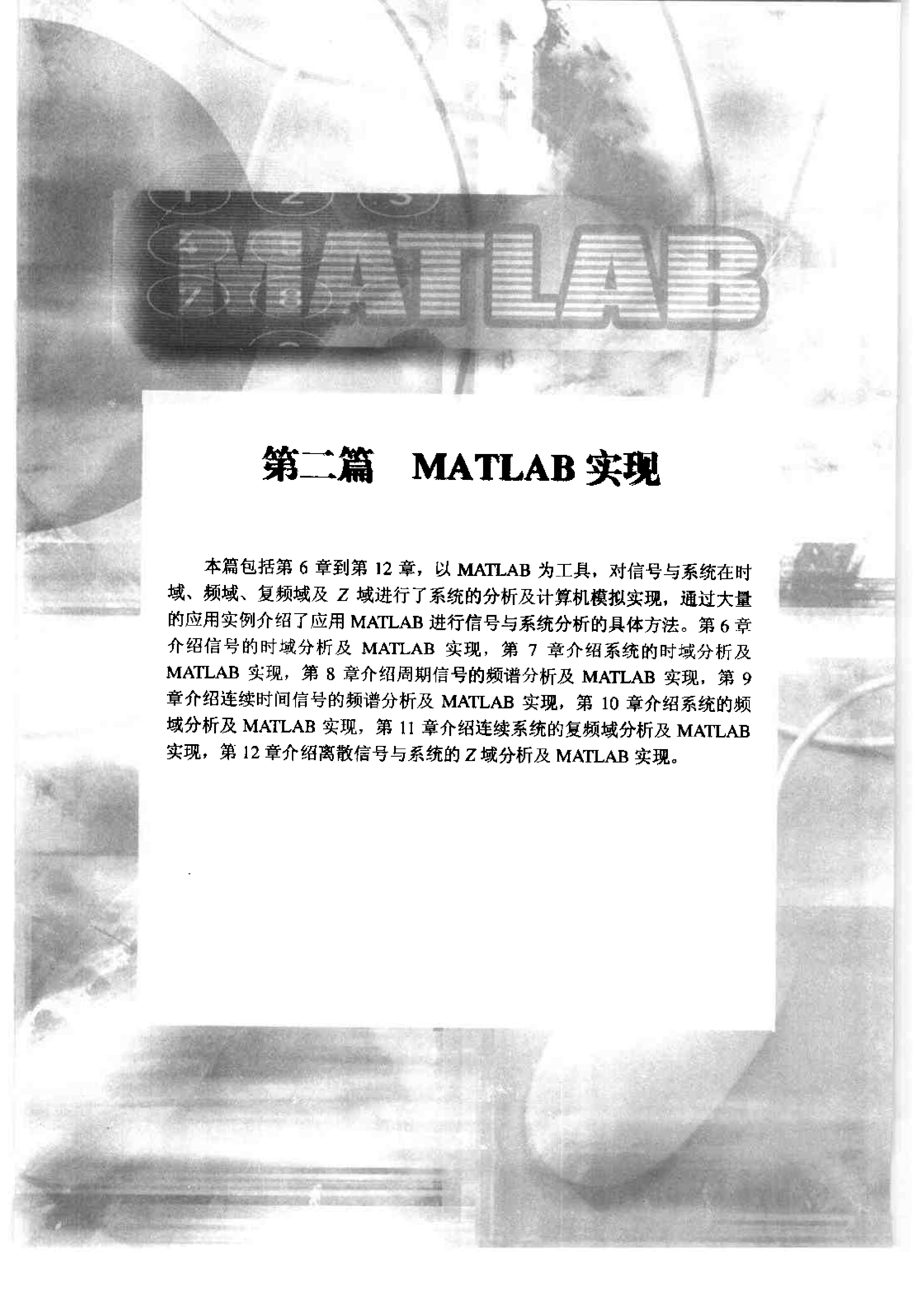


图 5-26 曲线对象设置属性的结果



日期：__________ 进程：__________ 备注：__________

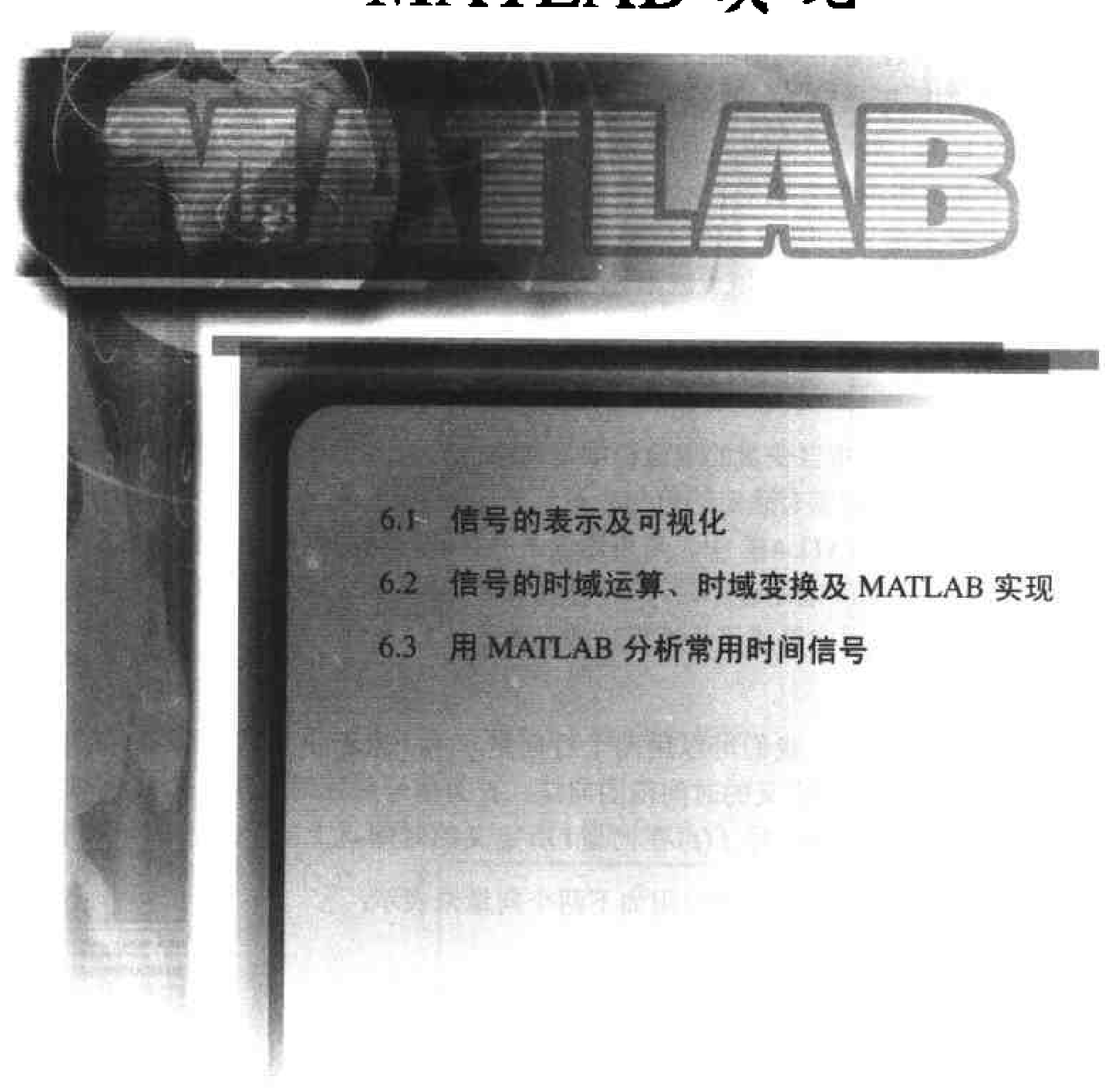


第二篇　MATLAB 实现

本篇包括第 6 章到第 12 章，以 MATLAB 为工具，对信号与系统在时域、频域、复频域及 Z 域进行了系统的分析及计算机模拟实现，通过大量的应用实例介绍了应用 MATLAB 进行信号与系统分析的具体方法。第 6 章介绍信号的时域分析及 MATLAB 实现，第 7 章介绍系统的时域分析及 MATLAB 实现，第 8 章介绍周期信号的频谱分析及 MATLAB 实现，第 9 章介绍连续时间信号的频谱分析及 MATLAB 实现，第 10 章介绍系统的频域分析及 MATLAB 实现，第 11 章介绍连续系统的复频域分析及 MATLAB 实现，第 12 章介绍离散信号与系统的 Z 域分析及 MATLAB 实现。



第6章　信号的时域分析及MATLAB实现





6.1 信号的表示及可视化

信号是消息的载体，是消息的一种表现形式。信号可以是多种多样的，通常表现为随时间变化的某些物理量，一般用 $f(t)$ 或 $f(k)$ 来表示。信号按照自变量的取值是否连续可分为连续时间信号和离散时间信号。

对信号进行时域分析，首先就需要将信号随时间变化的规律用二维曲线表示出来。对于简单的信号，我们可以通过手工来绘制其波形。但对于复杂的信号，手工绘制信号波形则显得十分困难，且难以绘制精确的曲线。

MATLAB 强大的图形处理功能及符号运算功能，为我们实现信号的可视化及时域分析提供了强有力的工具。本章将介绍如何用 MATLAB 来实现信号的可视化及时域分析。

在 MALAB 中通常用两种方法来表示信号，一种是用向量来表示信号，另一种则是用符号运算的方法来表示信号。用适当的 MATLAB 语句表示出信号后，我们就可以利用 MATLAB 的绘图命令绘制出直观的信号波形。

6.1.1 连续时间信号

所谓连续时间信号，是指自变量的取值范围是连续的，且对于一切自变量的取值，除了有若干不连续点以外，信号都有确定的值与之对应的信号。从严格意义上来讲，MATLAB 并不能处理连续信号，在 MATLAB 中，是用连续信号在等时间间隔点的样值来近似地表示连续信号的，当取样时间间隔足够小时，这些离散的样值就能较好地近似出连续信号。在 MATLAB 中连续信号可用向量或符号运算功能来表示。

1. 向量表示法

对于连续时间信号 $f(t)$，我们可以用两个行向量 f 和 t 来表示，其中向量 t 是形如 $t=t_1:p:t_2$ 的 MATLAB 命令定义的时间范围向量，t_1 为信号起始时间，t_2 为终止时间，p 为时间间隔。向量 f 为连续信号 $f(t)$ 在向量 t 所定义的时间点上的样值。例如对于连续信号 $f(t)=Sa(t)=\dfrac{\sin(t)}{t}$，我们可以用如下两个向量来表示：

```
t=-10:1.5:10
f=sin(t)./t
```

命令执行结果为：

```
t =
  Columns 1 through 7
  -10.0000   -8.5000   -7.0000   -5.5000   -4.0000   -2.5000   -1.0000
  Columns 8 through 14
    0.5000    2.0000    3.5000    5.0000    6.5000    8.0000    9.5000
t =
  Columns 1 through 7
  -0.0544    0.0939    0.0939   -0.1283   -0.1892    0.2394    0.8415
```



```
Columns 8 through 14
  0.9589    0.4546   -0.1002   -0.1918    0.0331    0.1237   -0.0079
```

用上述向量对连续信号进行表示后，就可以用 plot 命令来绘出该信号的时域波形。plot 命令可将点与点间用直线连接，当点与点间的距离很小时，绘出的图形就成了光滑的曲线。MATLAB 命令如下：

```
plot(t,f)
title('f(t)=Sa(t)')
xlabel('t')
axis([-10,10,-0.4,1.1])
```

绘制的信号波形如图 6-1 所示，当把时间间隔 p 取得更小（例如为 0.02）时，就可得到 $Sa(t)$ 较好的近似波形，如图 6-2 所示。

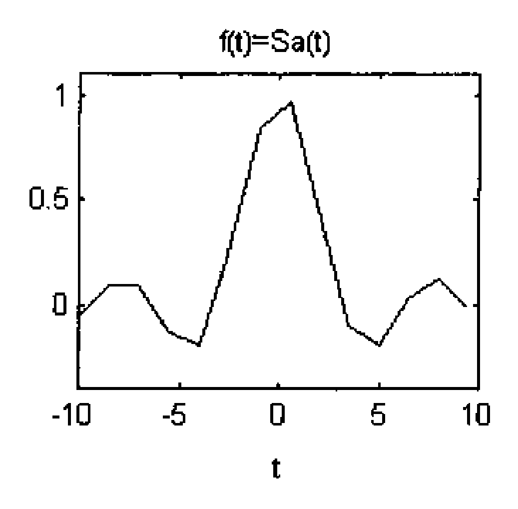


图 6-1　Sa(t)近似波形一

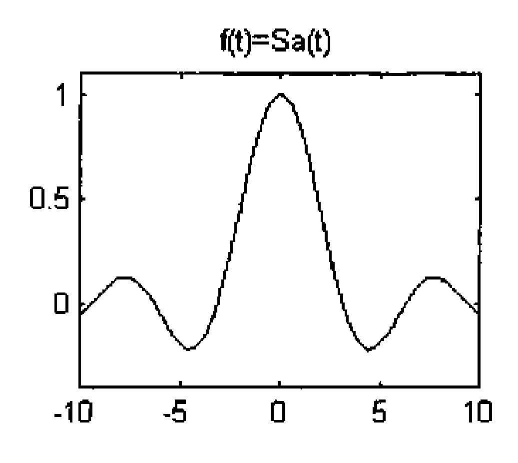


图 6-2　Sa(t)近似波形二

2．符号运算表示法

如果信号可以用一个符号表达式来表示它，则我们可用第 4 章介绍的 ezplot 命令绘制



出信号的波形。例如对于连续信号 $f(t)=\sin(\frac{\pi}{4}t)$，我们可以用符号表达式表示为：

```
f=sym('sin(pi/4*t)')
f =
sin(pi/4*t)
```

然后用 ezplot 命令绘制其波形：

```
ezplot(f,[-16,16])
```

该命令绘制的信号波形如图 6-3 所示

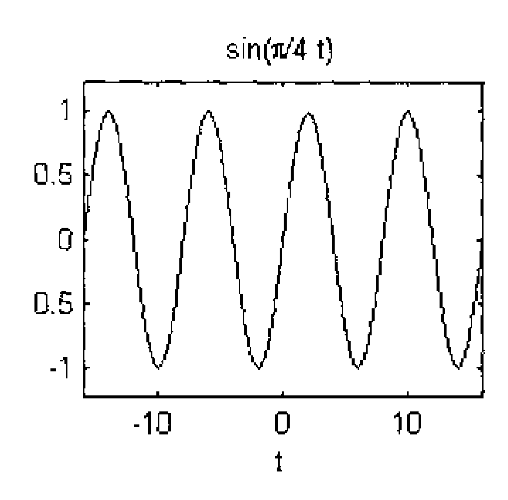


图 6-3　正弦信号波形图

下面就以几个常用的连续时间信号为例，说明如何用 MATLAB 来实现连续时间信号的可视化。

- 单位阶跃信号

单位阶跃信号的定义为：$\varepsilon(t)=\begin{cases}1 & t\geq 0\\ 0 & t<0\end{cases}$

单位阶跃信是信号分析的基本信号之一，在信号与系统分析中有着十分重要的作用，常用于简化信号的时域表示。根据单位阶跃信号的特性，我们可以用它来表示时限信号和单边信号（因果信号）。例如，我们可用两个不同延迟的单位阶跃信号来表示矩形门信号，即：$g(t)=\varepsilon(t+\tau)-\varepsilon(t-\tau)$

一种得到单位阶跃信号的方法是在 MATLAB 的 Symbolic Math Toolbox 中调用单位阶跃函数 Heaviside，这样可方便地表示出单位阶跃信号。但是，在用函数 ezplot 实现其可视化时，就出现一个问题：函数 ezplot 只能画出既存在于 Symbolic Math 工具箱中，又存在于总 MATLAB 工具箱中的函数，而 Heaviside 函数仅存在 Symbolic Math Toolbox 中，因此，就需要在自己的工作目录 work 下创建 Heaviside 的 M 文件，该文件如下：

```
function f=Heaviside(t)
f=(t>0);                    %t>0 时 f 为 1，否则为 0
```

正确定义出该函数并保存运行后，就可调用该函数了。如先定义向量：



```
t=-1:0.01:3
```

然后调用 Heaviside 函数表示出该信号并绘出波形

```
f=heaviside(t)
plot(t,f)
axis([-1,3,-0.2,1.2])
```

得到波形如图 6-4 所示。

例 6-1：用 MATLAB 画出信号 $f(t)=\varepsilon(t+3)-2\varepsilon(t)$ 的波形。

解：用下列 MATLAB 命令即可实现 $f(t)$ 的表示和绘制：

```
f=sym('heaviside(t+3)-2*heaviside(t)')
        ezplot(f,[-5,4]),
hold on,plot([0,0],[-1,1]),
axis([-5,4,-1.1,1.1]),hold off
```

命令执行后，绘出波形如图 6-5 所示。

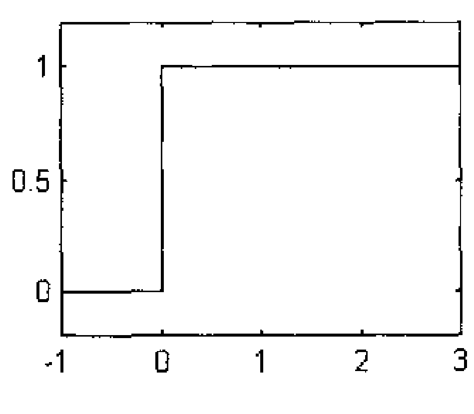


图 6-4　单位阶跃信号

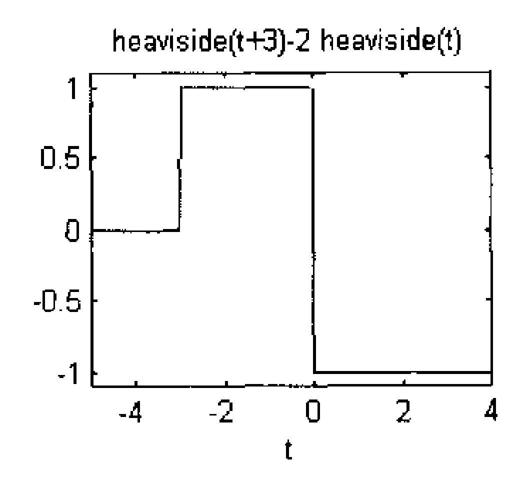


图 6-5　连续信号波形

另一种表示单位阶跃信号的方法是用向量 f 和 t 分别表示信号的样值及对应时刻值。例如　f=[0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1]

t=[-1,-0.9,-0.8,-0.7,-0.6,-0.5,-0.4,-0.3,-0.2,-0.1,0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1]

零时刻以前，信号样值为零，从零时刻起，信号样值为一，定义出这样的两个向量后，就可用 plot 命令绘出波形图了。

下面就是表示和绘制单位阶跃信号的子程序，调用它不仅能画出单位阶跃信号 $\varepsilon(t)$ 的波形，还可画出单位阶跃信号沿时间轴平移时间 t_0 的波形图(即 $\varepsilon(t+t_0)$ 的波形)。其中 $t1,t2$ 表示信号的起始时刻，$t0$ 表示信号沿坐标的平移量，且 $t1 \leqslant t0 \leqslant t2$，$t0>0$ 时向左平移，$t0<0$ 时向右平移。

```
%t0 时刻以前信号为零，在 t0 处有一跃变，以后为一
function jieyao(t1,t2,t0)
t=t1:0.01:-t0;                    %t0 时刻前时间样本向量
tt=-t0:0.01:t2;                   %t0 时刻后时间样本向量
n=length(t);                      %t0 前时间样本点向量长度
nn=length(tt)                     %t0 后样本点向量长度
u=zeros(1,n);                     %t0 前各样本点信号值赋值为零
uu=ones(1,nn);                    %t0 后各样本点信号值赋值为一
plot(tt,uu)                       %绘出 t0 时刻后波形
hold on                           %允许在同一坐标系中添加图形
plot(t,u)                         %绘出 t0 时刻前波形
plot([-t0,-t0],[0,1])             %添加直线
hold off                          %关闭添加命令
title('单位阶跃信号')              %图形标题
axis([t1,t2,-0.2,1.5])            %限制坐标范围
```

现在，我们就调用 jieyao 函数，绘出 $\varepsilon(t)$，$-1 \leqslant t \leqslant 4$ 的波形，MATLAB 的调用命令为：

```
jieyao(-1,4,0)
```

程序执行后绘出波形如图 6-6 所示：

第三种方法是用符号函数 $\operatorname{sgn}(t)=\begin{cases}1 & t>0\\ -1 & t<0\end{cases}$ 来生成单位阶跃函数，即 $\varepsilon(t)=1/2+(1/2)\operatorname{sgn}(t)$。而 $\operatorname{sgn}(t)$ 的表示可调用 MATLAB 中的符号函数 sign 来实现。

若定义向量：

```
t=-5:0.05:5
f=sign(t)
```

然后，用下面的命令就可绘出符号函数的波形，如图 6-7 所示。

```
plot(t,f),axis([-5,5,-1.1,1.1])
```



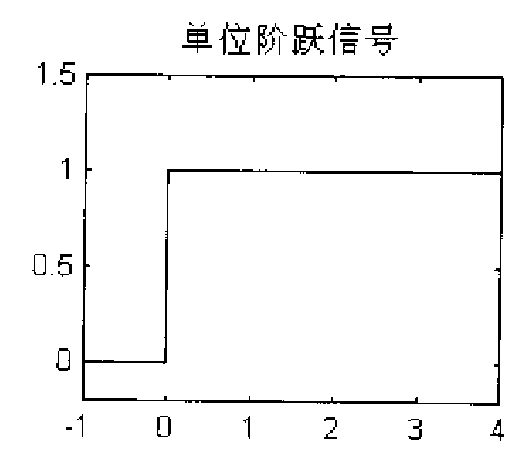


图 6-6　单位阶跃信号波形

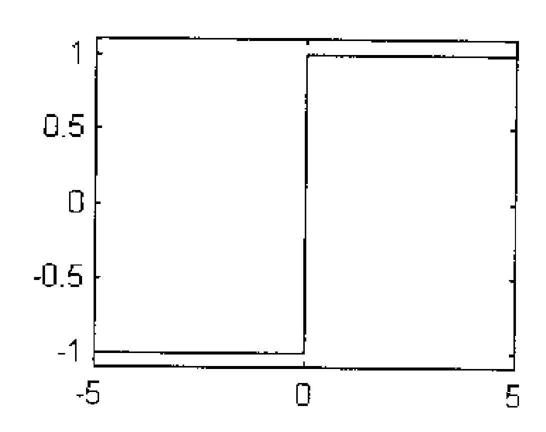


图 6-7　符号函数波形

再用以下命令表示单位阶跃信号并绘出它的波形，如图 6-8 所示。

```
ff=1/2+1/2*f
plot(t,ff),axis([-5,5,-0.1,1.1])
```

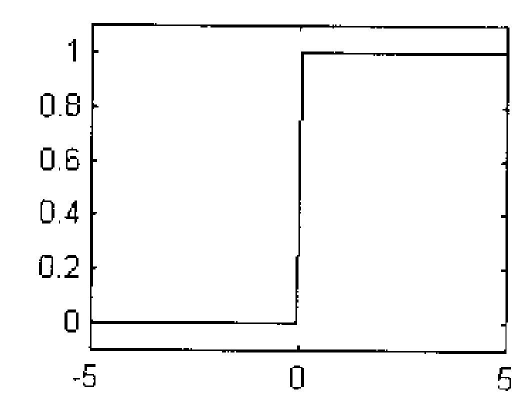
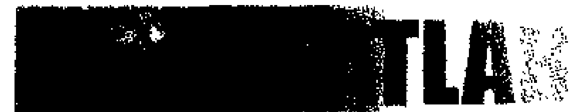


图 6-8　单位阶跃信号波形



- 单位冲击信号 $\delta(t)$

单位冲击信号的定义为 $\begin{cases}\int_{-\infty}^{\infty}\delta(t)\mathrm{d}t=1\\ \delta(t)=0,t\neq 0\end{cases}$

$\delta(t)$ 的定义表明，该信号除原点以外，处处为零，且信号面积为一。即设有门函数 $\delta_{\Delta}(t)$，则由 $\delta(t)$ 定义可得：

$$\delta(t)=\lim_{\Delta\to 0}\delta_{\Delta}(t)$$

$\delta(t)$是信号与系统分析的基本信号之一，是我们进行信号分析的基础。

严格说来，MATLAB 是不能表示单位冲击信号的，但我们可用时间宽度为 $\mathrm{d}t$，高度为 $1/\mathrm{d}t$ 的矩形脉冲来近似地表示冲击信号。当 $\mathrm{d}t$ 趋近于零时，就较好地近似出冲击信号的实际波形。下面是绘制单位冲击信号及其在时间轴上的平移信号 $\delta(t+t_0)$ 的 MATLAB 子程序，其中 $t1$，$t2$ 表示信号的起始时刻，$t0$ 表示信号沿坐标的平移量，$t0>$时向左平移，$t0<0$ 时向右平移。绘图命令用 stairs，该命令一般用于绘制类似楼梯形状的步进图形，在这里使用该命令是因为显示连续信号中的不连续点用 stairs 命令绘图效果较好。执行后果如图 6-9 所示。

```
function chongji(t1,t2,t0)
dt=0.01;                        %信号时间间隔
t=t1:dt:t2;                     %信号时间样本点向量
n=length(t);                    %时间样本点向量长度
x=zeros(1,n);                   %各样本点信号值赋值为零
x(1,(-t0-t1)/dt+1)=1/dt;        %在时间 t=-t0 处，给样本点赋值为 1/dt
stairs(t,x);
axis([t1,t2,0,1.2/dt])
title('单位冲击信号 δ(t)')
```

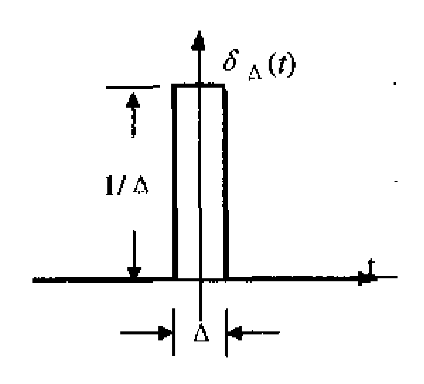


图 6-9 门信号 $\delta_{\Delta}(t)$

下面就调用 chongji 函数绘制 $\delta(t)$,-1≤t≤5 的波形。MATLAB 调用命令为：chongji(-1,5,0)程序执行后，MATLAB 自动画出如图 6-10 所示的波形。

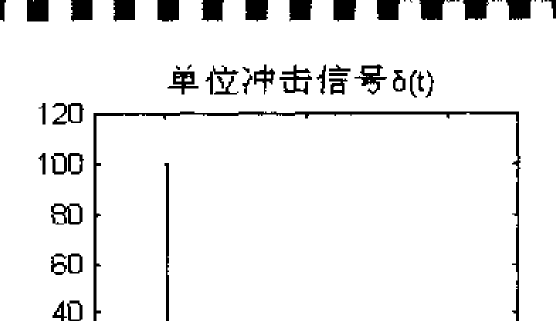


图 6-10　单位冲击信号波形

例 6-2：用 MATLAB 命令绘制单边指数信号 $e^{-1.5t}\varepsilon(t)$ 在时间 $0 \le t \le 3$ 区间的波形。

解：我们可用两向量 f 和 t 来表示信号，用 plot 命令绘制其波形，具体命令如下：

```
t=0:0.05:3;
f=exp(-1.5*t);
plot(t,f)
axis([0,3,0,1.2])
title('单边指数信号')
text(3.1,0.05,'t')
```

程序执行后，产生如图 6-11 所示的波形。

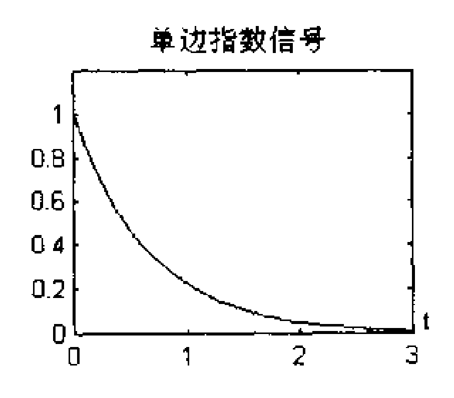


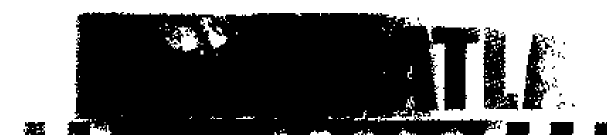
图 6-11　单边指数信号波形

6.1.2　离散时间信号

一般说来，离散时间信号用 $f(k)$ 表示，其中变量 k 为整数，代表离散的采样时间点。

$f(k)$ 可表示为：

$$f(k)=\{\cdots f(-2),f(-1),f(0),f(1),f(2)\cdots\}$$





k=0

在 MATLAB 中，用一个向量 f 即可表示一个有限长度的序列。但是，这样的向量并没有包含其对应的时间序号信息。所以，要完整地表示离散信号需要用两个向量。

如序列：　$f(k)=\{1,2,-1,3,2,4,-1\}$
$\uparrow$ k=0

在 MATLAB 中应表示为：

$k=[-3,-2,-1,0,1,2,3]$ 或是 $k=-3:3$；　$f=[1,2,-1,3,2,4,-1]$；

在用 MATLAB 表示离散序列并将其可视化时，我们要注意以下几点：第一，与连续时间信号不同，离散时间信号无法用符号运算来表示；第二，由于在 MATLAB 中，矩阵的元素个数是有限的，因此，MATLAB 无法表示无限序列；第三，在绘制离散信号波形时，要使用专门绘制离散数据的 stem 命令，而不是 plot 命令。如对于上面定义的二向量 f 和 k，可用如下 stem 命令绘图：

```
stem(k,f,'filled'), axis([-4,4,-1.5,4.5])
```

得到对应的序列波形图，如图 6-12 所示。

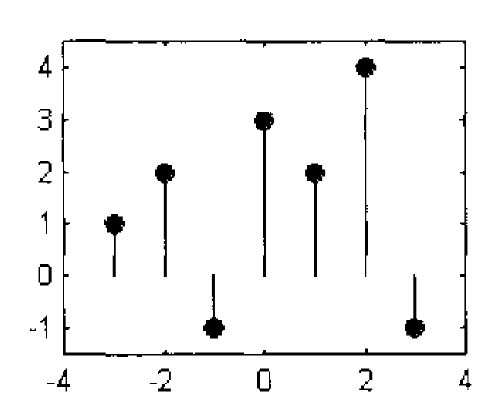


图 6-12　离散序列波形

下面通过一些常用的离散信号来说明如何用 MATLAB 来实现离散序列的表示和可视化。

单位序列 $\delta(k)$

单位序列的定义为 $\delta(k)=\begin{cases}1 & k=0\\0 & k\neq 0\end{cases}$

由单位序列的定义知，只有在 $k=0$ 时，$\delta(k)$ 的值为 1，而当 $k\neq 0$ 时 $\delta(k)$ 的值为零，因此，用 MATLAB 来表示单位序列及绘制其波形非常简单。

下面是 MATLAB 绘制单位序列 $\delta(k+k_0)$ 的子程序，其中 $k0$ 为 $\delta(k)$ 在时间轴上的位移量，$k0<0$ 则右移，$k0>0$ 则左移，$k1$，$k2$ 为时间序列的起始时间序号，且 $k1\leqslant k0\leqslant k2$，调用该函数就可以绘出单位序列及其移位序列的波形图：



```
function dwxulie(k1,k2,k0)
k=k1:k2;
n=length(k);
f=zeros(1,n);
f(1,-k0-k1+1)=1;                                  %在 k0 时刻，信号赋值为 1
stem(k,f,'filled')
axis([k1,k2,0,1.5])
title('单位序列 δ(k)')
```

例 6-3：画出 $\delta(k)$ 在-5≤k≤5 区间的波形。

解：调用上述程序来完成，调用命令为：

```
dwxulie(-5,5,0)
```

程序执行后产生图 6-13 所示的波形。

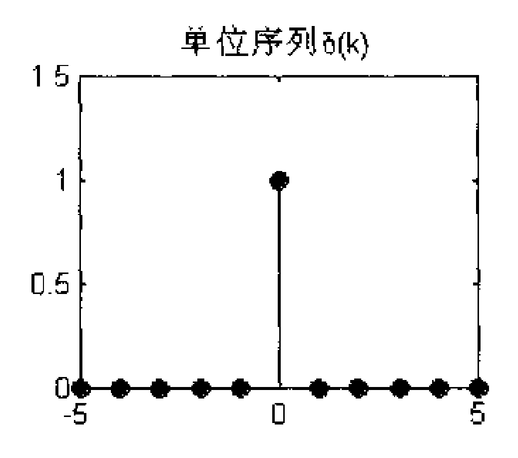


图 6-13　单位序列波形

2．单位阶跃序列 $\varepsilon(k)$

$$\text{单位阶跃序列的定义为}\ \varepsilon(k)=\begin{cases}1 & k\geqslant 0\\ 0 & k<0\end{cases}$$

类似单位序列，下面也给出绘制单位阶跃序列 $\varepsilon(k+k_0)$（k0，k1，k2 的含义同上例）的 MATLAB 子程序：

```
function jyxulie(k1,k2,k0)
k=k1:-k0-1;
kk=-k0:k2;
n=length(k);
nn=length(kk)
u=zeros(1,n); %k0 前信号赋值为零
uu=ones(1,nn);%k0 后信号赋值为一
```

```
stem(kk,uu,'filled')
hold on
stem(k,u,'filled')
hold off
title('单位阶跃序列')
axis([k1 k2 0 1.5])
```

例 6-4：用 MATLAB 绘出单位阶跃序列 $\varepsilon(k)$ 在 $-3 \leqslant k \leqslant 8$ 区间的波形。

解：可调用上述程序来完成，调用命令为：

```
jyxulie(-3,8,0)
```

程序执行后产生图 6-14 所示的波形。

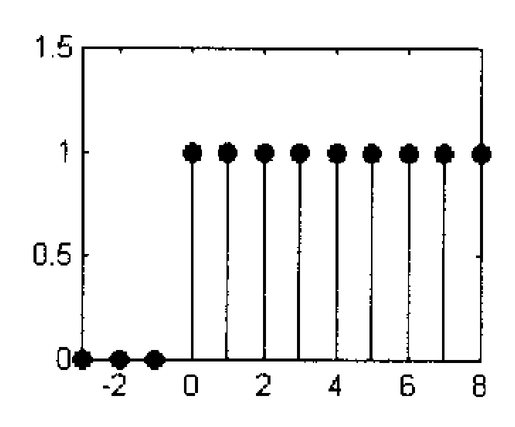


图 6-14　单位阶跃序列波形

本节使用了 stairs、plot、stem 和 ezplot 四个绘图命令，显示连续信号中的不连续点用 stairs 命令绘图效果较好；在绘制连续信号时要得到光滑曲线用 plot 命令；绘制离散信号则要用 stem 命令；绘制用 MATLAB 符号表达式表示的信号用 ezplot 命令。

6.2　信号的时域运算、时域变换及 MATLAB 实现

信号的时域运算包括信号的相加、相乘，信号的时域变换包括信号的平移、反折、倒相及信号的尺度变换。下面就分别介绍连续时间信号和离散时间信的各种时域运算、变换及 MATLAB 实现。

6.2.1　连续信号的时域运算与时域变换

如前所述，MATLAB 可以有两种方法来表示连续信号。用这两种方法均可实现连续信号的时域运算和变换，但用符号运算的方法则较为简便。下面就分别介绍各种运算、变换的符号运算实现方法。



1. 相加

连续信号的相加，是指两信号的对应时刻值相加，即：

$$f(t)=f_1(t)+f_2(t)$$

下面用 MATLAB 的符号运算命令来表示两连续信号的相加，然后用 ezplot 命令绘制出其结果波形图。其中 *f*1，*f*2 是两个用符号表达式表示的连续信号，s 为相加得到的和信号的符号表达式。

```
s=symadd(f1,f2)或 s=f1+f2
ezplot(s)
```

2. 相乘

连续信号的相乘，是指两信号的对应时刻值相乘，即：

$$f(t)=f_1(t)\times f_2(t)$$

与相加运算类似，我们用下面的 MATLAB 命令来实现连续信号的相乘及其结果的可视化，其中 *f*1，*f*2 为两个用符号表达式表示的信号，*w* 为相乘得到的积信号的符号表达式。

```
w=symmul(f1,f2) 或 w=f1*f2
ezplot(w)
```

3. 移位

连续信号的移位也称平移。对于连续信号 $f(t)$，若有常数 $t_0>0$，延时信号 $f(t-t_0)$ 是将原信号沿正 t 轴方向平移时间 t_0，而 $f(t+t_0)$ 是将原信号沿负 t 轴方向移动时间 t_0。我们可用下面的命令来实现连续信号的平移及其结果的可视化，其中 f 是用符号表达式表示的连续时间信号，t 是符号变量，subs 命令则将连续信号中的时间变量 t 用 *t-t0* 替换：

```
y=subs(f,t,t-t0);
ezplot(y)
```

4. 反折

连续信号的反折，是指将信号以纵坐标为轴反折，即将信号 $f(t)$ 中的自变量 t 换为 $-t$。与连续信号的平移类似，我们用下面的命令实现连续信号的反折及其结果的可视化，其中 f 是用符号表达式表示的连续时间信号，t 是符号变量：

```
y=subs(f,t,-t)
ezplot(y)
```

5. 尺度变换

连续信号的尺度变换，是指将信号的横坐标进行展宽或压缩变换，即将信号 $f(t)$ 中的自变量 t 换为 at，当 $a>1$ 时，信号 $f(at)$ 以原点为基准，沿横轴压缩到原来的 $1/a$；当 $0<a<1$ 时，信号 $f(at)$ 将沿横轴展宽至原来的 $1/a$ 倍。我们用下面的命令来实现连续信号的尺度变换及其结果的可视化，其中 f 是用符号表达式表示的连续时间信号，t 是符号变量：




```
y=subs(f,t,a*t)
ezplot(y)
```

6．倒相

连续信号的倒相，是指将信号 $f(t)$ 以横轴为对称轴对折得到 $-f(t)$，可用下面的命令实现连续信号的倒相及其结果的可视化，其中 f 是用符号表达式表示的连续时间信号。

```
y=-f
ezplot(y)
```

对于以上的命令，可在画图命令之后加入坐标轴的调整等命令，以使画出的图形更清晰、直观。

下面举例说明如何用 MATLAB 来实现连续信号的时域运算、变换及其结果的可视化。

例 6-5：设信号 $f(t)=(1+\frac{t}{2})\times[\varepsilon(t+2)-\varepsilon(t-2)]$，用 MATLAB 求 $f(t+2)$，$f(t-2)$，$f(-t)$，$f(2t)$，$-f(t)$，并绘出其时域波形。

解：根据前面的介绍，我们可用符号运算来实现上述过程，MATLAB 命令如下：

```
syms t
f=sym('(t/2+1)*(heaviside(t+2)-heaviside(t-2))')
subplot(2,3,1),ezplot(f,[-3,3])
y1=subs(f,t,t+2)
subplot(2,3,2),ezplot(y1,[-5,1])
y2=subs(f,t,t-2)
subplot(2,3,3),ezplot(y2,[-1,5])
y3=subs(f,t,-t)
subplot(2,3,4),ezplot(y3,[-3,3])
y4=subs(f,t,2*t)
subplot(2,3,5),ezplot(y4,[-2,2])
y5=-f
subplot(2,3,6),ezplot(y5,[-3,3])
```

命令执行后得到 f,y1,y2,y3,y4,y5 的符号表达式如下：

```
f =
(t/2+1)*(heaviside(t+2)-heaviside(t-2))
y1 =
(1/2*t+2)*(heaviside(t+4)-heaviside(t))
y2 =
1/2*t*(heaviside(t)-heaviside(t-4))
y3 =
(-1/2*t+1)*(heaviside(-t+2)-heaviside(-t-2))
y4 =
(t+1)*(heaviside(2*t+2)-heaviside(2*t-2))
```



```
y5 =
(1/2*t+1)*(heaviside(t+2) heaviside(t-2))
```

绘制的信号时域变换波形如图 6-15 所示。

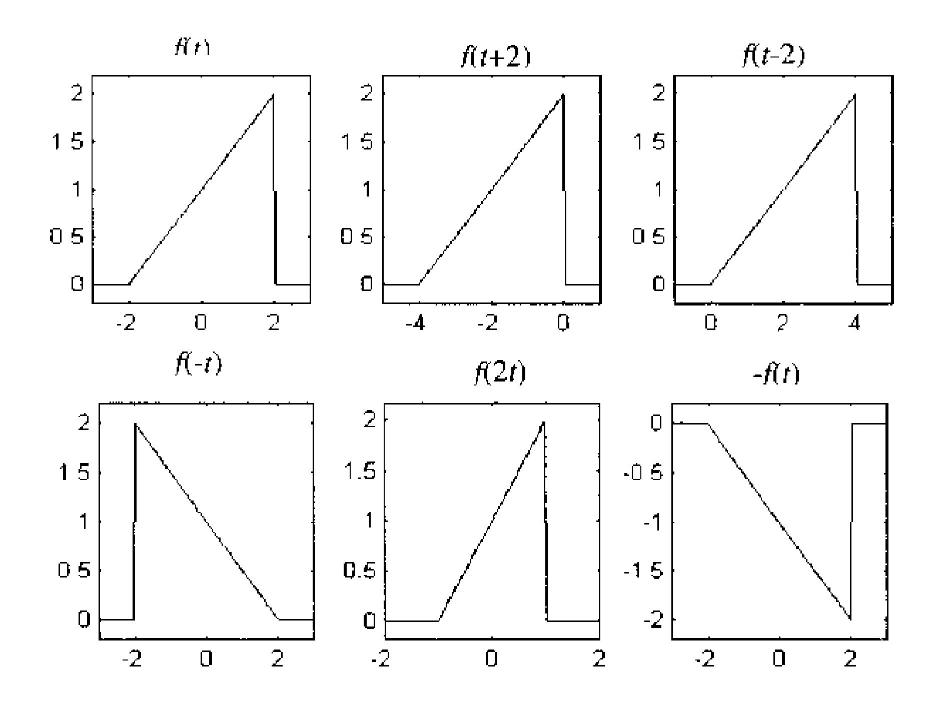


图 6-15　信号的时域变换波形图

例 6-6：已知如右图所示信号 $f_1(t)$ 及信号 $f_2(t)=\sin(2\pi t)$，用 MATLAB 绘出满足下列要求的信号波形。

(1) $f_3(t)=f_1(-t)+f_1(t)$　　(2) $f_4(t)=-[f_1(-t)+f_1(t)]$

(3) $f_5(t)=f_2(t)\times f_3(t)$　　(4) $f_6(t)=f_1(t)\times f_2(t)$

解：

信号 $f_1(t)$ 可表示为 $f_1(t)=(-t+4)[\varepsilon(t)-\varepsilon(t-4)]$，这样我们即可用 MATLAB 的符号运算功能来实现题目要求的时域运算：

```
syms t
f1=sym('(-1*t+4)*(heaviside(t)-heaviside(t-4))')
subplot(2,3,1),ezplot(f1)
f2=sym('sin(2*pi*t)')
subplot(2,3,4),ezplot(f2,[-4,4])
y1=subs(f1,t,-t)
f3=f1+y1
subplot(2,3,2),ezplot(f3)
f4=-f3
```

```
subplot(2,3,3),ezplot(f4)
f5=f2*f3
subplot(2,3,5),ezplot(f5)
f6=f1*f2
subplot(2,3,6),ezplot(f6)
```

绘制的信号时域波形如图 6-16 所示：

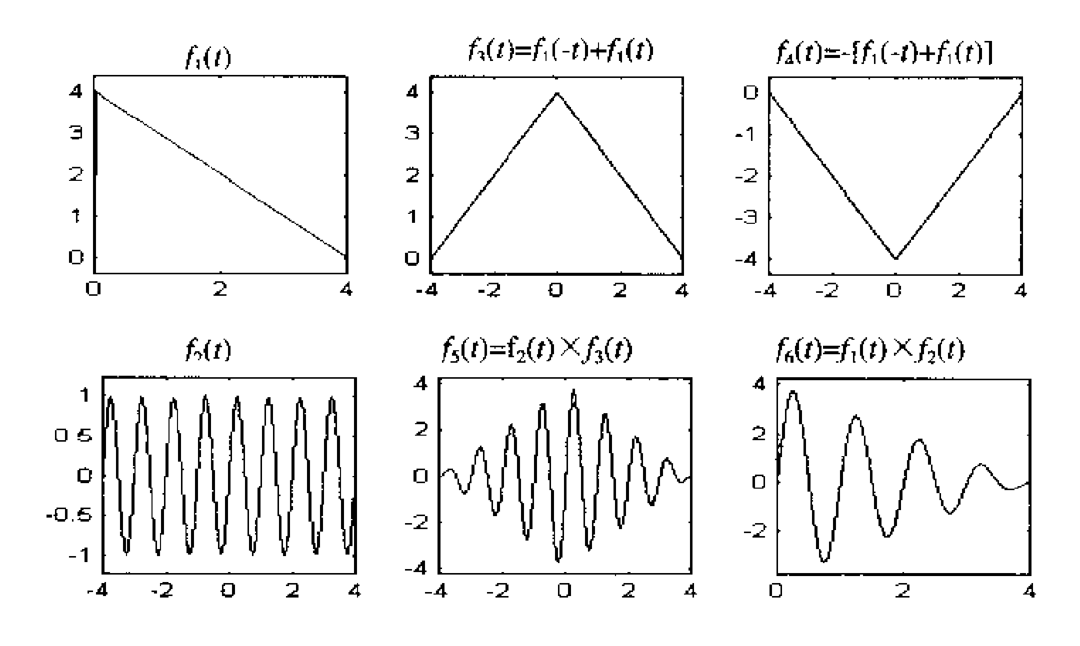


图 6-16 例 6-6 的信号波形图

6.2.2 离散序列的时域运算及时域变换

对于离散序列来说，序列相加、相乘是将两序列对应时间序号的值逐项相加或相乘，平移、反折、及倒相变换与连续信号的定义完全相同，这里就不再赘述。但需要注意，与连续信号不同的是，在 MATLAB 中，离散序列的时域运算和变换不能用符号运算来实现，而必须用向量表示的方法，即在 MATLAB 中离散序列的相加、相乘需表示成两个向量的相加、相乘，因而参加运算的两序列向量必须具有相同的维数。

下面是实现离散序列相加、相乘的实用子程序及实例。

1. 离散序列相加及其结果可视化的实现

在该函数中，将要进行相加运算的二序列向量 *f1*、*f2* 通过补零的方式成为同维数的二序列向量 *s1*、*s2*，因而在调用该函数时，要进行相加运算的二序列向量维数可以不同：

```
function[f,k]=lsxj(f1,f2,k1,k2)
%实现 f(k)=f1(k)+f2(k)，f1,f2,k1,k2 是参加运算的二离散序列及其对应的时间序列向量，f 和 k 为返回的和序列及其对应的时间序列向量
k=min(min(k1),min(k2)):max(max(k1),max(k2));          %构造和序列的长度
s1=zeros(1,length(k));s2=s1;                          %初始化新向量
```

```
s1(find((k>=min(k1))&(k<=max(k1))==1))=f1;                    %将 f1 中在和序列范围内但又无定义的点赋值为零
s2(find((k>=min(k2))&(k<=max(k2))==1))=f2;                    %将 f2 中在和序列范围内但又无定义的点赋值为零
f=s1+s2;                                                      %两长度相等序列求和
stem(k,f,'filled')
axis([(min(min(k1),min(k2))-1),(max(max(k1),max(k2))+1),(min(f)-0.5),(max(f)+0.5)])
                                                              %坐标轴显示范围
```

例 6-7：有两离散序列，用 MATLAB 绘出它们的波形及 $f_1(k)+f_2(k)$ 的波形。

$$f_1(k)=\{-2,-1,\underset{\uparrow k=0}{0},1,2\},\ f_2(k)=\{1,\underset{\uparrow k=0}{1},1\}$$

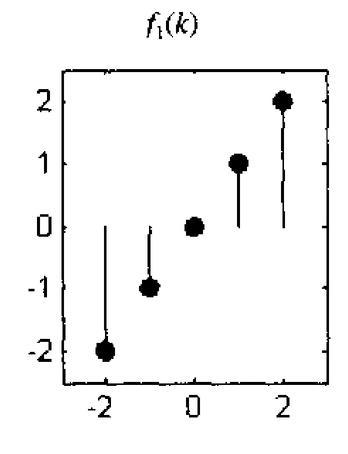
解：

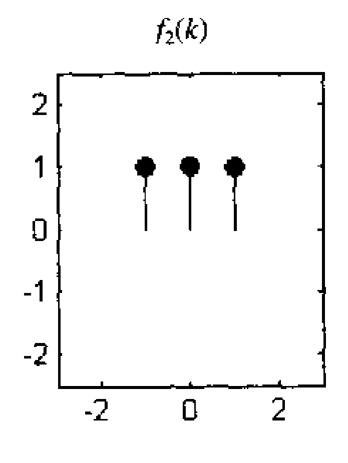
MATLAB 命令为：

```
f1=-2:2;
k1=-2:2;
f2=[1 1 1];
k2=-1:1;
stem(k1,f1),axis(-3,3,-2.5,2.5)
stem(k2,f2),axis(-3,3,-2.5,2.5)
[f,k]=lsxj(f1,f2,k1,k2)
```

程序运行结果如下，绘制的波形如图 6-17 所示：

```
f=
-2    0    1    2    2
k=
-2    -1    0    1    2
```





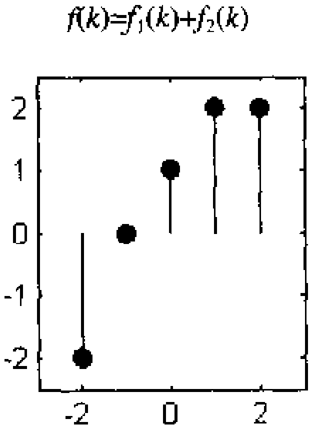


图 6-17　离散序列相加波形图



2．离散序列相乘及其结果可视化的实现

在该函数中，将要进行相乘运算的二序列向量 $f1$、$f2$ 通过补零的方式成为同维数的序列向量 $s1$、$s2$，因而在调用该函数时，要进行相乘运算的二序列向量维数可以不同：

```
function[f,k]=lsxc(f1,f2,k1,k2)
%实现 f(k)=f1(k)×f2(k)，f1,f2,k1,k2 是参加运算的二离散序列及其对应的时间序列向量，f 和 k 为返回的积序列及其对应的时间序列向量
k=min(min(k1),min(k2)):max(max(k1),max(k2));
s1=zeros(1,length(k));s2=s1;
s1(find((k>=min(k1))&(k<=max(k1))==1))=f1;
s2(find((k>=min(k2))&(k<=max(k2))==1))=f2;
f=s1.*s2;
stem(k,f,'filled')
axis([(min(min(k1),min(k2))-1),(max(max(k1),max(k2))+1),(min(f)-0.5),(max(f)+0.5)])
```

该程序的调用方法与上例同。

3．离散序列反折及 MATLAB 实现

离散序列的反折，即是将表示离散序列的两向量以零时刻的取值为基准点，以纵轴为对称轴反折，向量的反折可用 MATLAB 中的 fliplr 函数来实现。

下面就是用 MATLAB 来实现离散序列反折及其结果可视化的子函数：

```
function[f,k]=lsfz(f1,k1)
f=fliplr(f1);k=-fliplr(k1);                    %调用 fliplr 函数实现向量 f1 和 k1 的反折
stem(k,f,'filled')
axis([min(k)-1,max(k)+1, min(f)-0.5,max(f)+0.5])
```

$$f(k)=\begin{cases}2^k & -3\le k\le 3\\ 0 & k=\text{其他}\end{cases}$$

例 6-8：用 MATLAB 绘出离散序列的波形及反折后的波形。

解：

MATLAB 命令为：

```
k=-3:3
    f=2.^k
stem(k,f),axis([-4,4,-0.5,8.5])
lsfz(f,k)
```

命令执行后得到原序列波形图及反折后序列波形图 6-18。

4．离散序列的平移及 MATLAB 实现

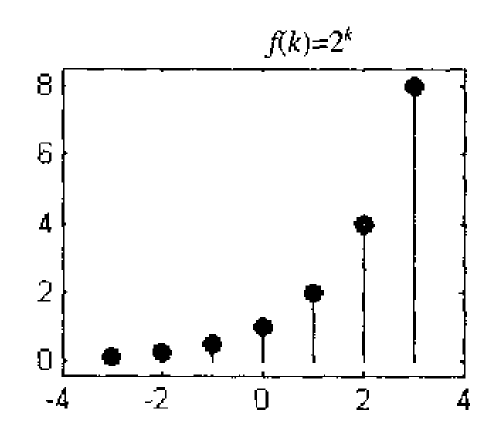
离散序列的平移可看作是将离散序列的时间序号向量平移，而表示对应时间序号点的序列样值不变，当序列向左移动 $k0$ 个单位时，所有时间序号向量都减小 $k0$ 个单位，反之则增加 $k0$ 个单位。可用下面的子函数来实现：

```
function[f,k]=lsyw(ff,kk,k0)
```



% ff,kk 是未平移前的离散序列及其对应的时间序列向量，k0 是平移量，f 和 k 为返回的平移之后的序列及其对应的时间序列向量

```
k=kk+k0;f=ff;
stem(k,f,'filled')
axis([min(k)-1,max(k)+1,min(f)-0.5,max(f)+0.5])
```



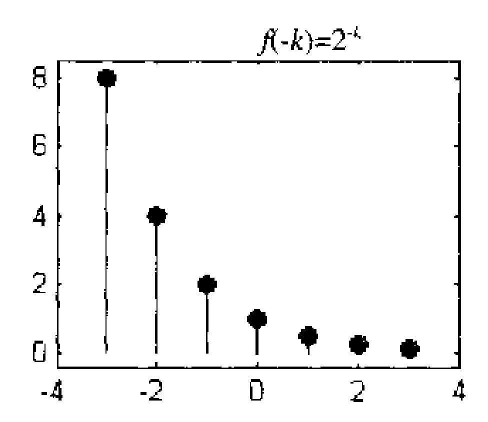


图 6-18　离散信号的反折

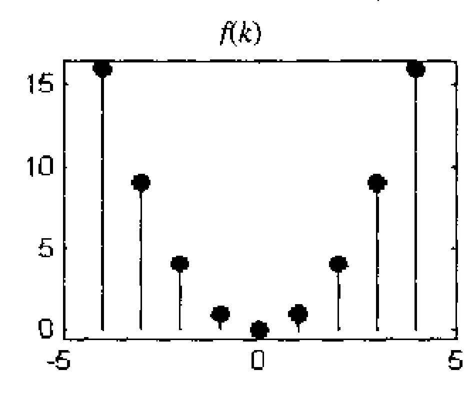
例 6-9：有 $f(k)=\begin{cases} k^2 & -4\leqslant k\leqslant 4 \\ 0 & k=\text{其他} \end{cases}$ 序列绘出 $f(k)$ 的波形及 $y=f(k-2)$ 的波形。

解：

MATLAB 命令为：

```
k=-4:4
f=k.^2
stem(k,f,'filled'),axis([-5,5,-0.5,16.5])
lsyw(f,k,2)
```

绘制的序列波形如图 6-19 所示。



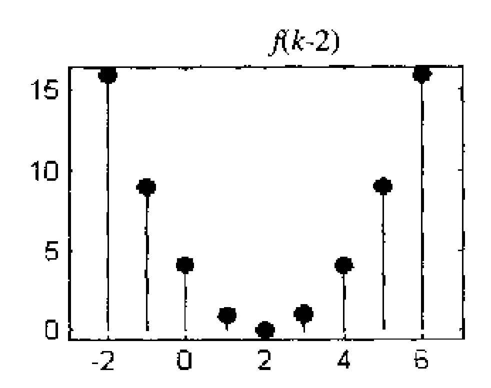


图 6-19　离散序列的反折

5. 离散序列的倒相变换及 MATLAB 实现

离散序列的倒相可看作是将表示序列样值的向量取反，而对应的时间序号向量不变得到的离散时间序列。可用下面的子函数来实现：

```
function[f, k]=lsdx(ff,kk)
% ff,kk 是未倒相前的离散序列及其对应的时间序列向量， f 和 k 为返回的倒相之后的序列及其对应的时间序列向量
f=-ff
k=kk
stem(k,f,'filled')
axis([min(k)-1,max(k)+1,min(f)-0.5,max(f)+0.5])
```

以上例中的信号 $f(k)$为例，用如下命令可得到它的倒相波形图，如图 6-20 所示。

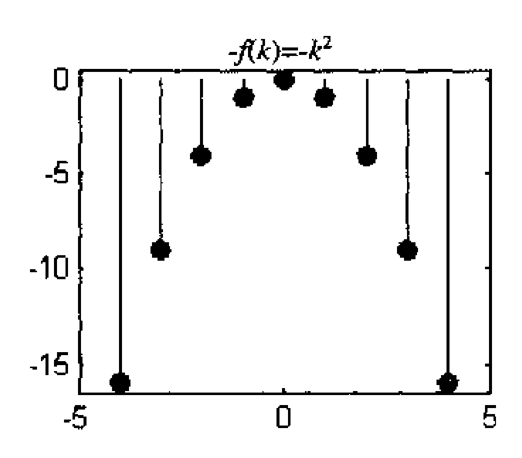


图 6-20　离散序列的倒相波形

```
k=-4:4
f=k.^2
lsdx(f,k)
```

6.3　用 MATLAB 分析常用时间信号

从上述 MATLAB 的运用我们可以看出，MATLAB 可以帮助我们快速、方便地绘制出信号的时域波形，这为我们分析信号的时域特性提供了极大的方便，也可使得分析过程变得更加直观。下面，我们用 MATLAB 来分析几个常用信号的时域特性。

6.3.1　连续时间信号

1. 正弦信号

正弦信号定义为：



$$f(t) = A\cos(\omega t + \varphi)$$

其中 A 为振幅，ω 为角频率，φ 为初相位。

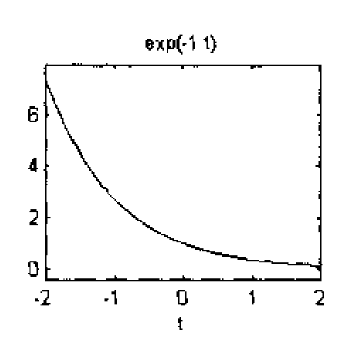
正弦信号的时域特性由振幅、角频率和初相位三个量确定。正弦信号的角频率 ω、频率 f 和周期 T 之间存在着如下关系：

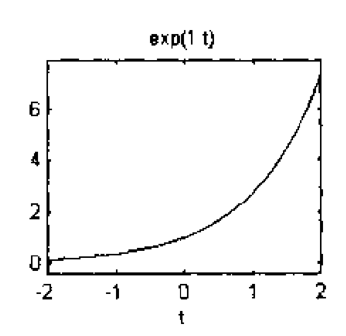
$$f = \frac{1}{T} \qquad \omega = 2\pi f$$

下面我们用 MATLAB 来绘制正弦信号 $f(t) = 3\sin(\omega t)$，当 $\omega = \frac{\pi}{2}$、$\omega = \pi$ 和 $\omega = \frac{3\pi}{2}$ 的时域波形，对应的 MATLAB 命令如下：

```
f=sym('3*sin((w)*t)')
f1=subs(f,'w','pi/2')
ezplot(f1,[0,4*pi])
f2=subs(f,'w','pi')
ezplot(f2,[0,4*pi])
f3=subs(f,'w','3*pi/2')
ezplot(f3,[0,4*pi])
```

绘制的正弦信号时域波形如图 6-21 所示。





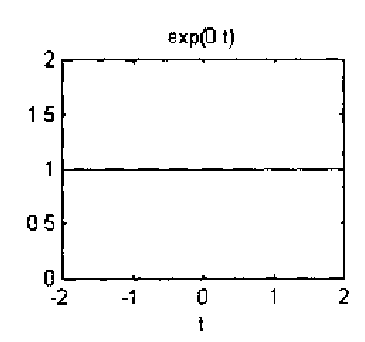


图 6-21　正弦时间信号

同理，我们可用相同的方法绘制出正弦信号不同初始相位情况下的时域波形。

实指数信号

实指数信号的一般形式为：

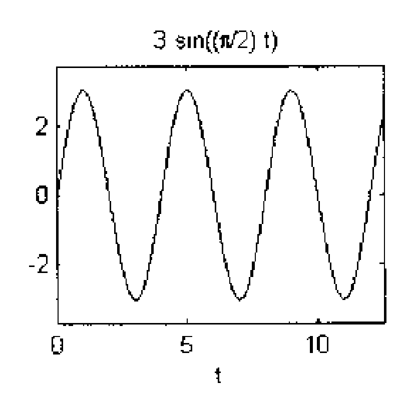
$f(t) = ce^{at}$，其中 c 和 a 为实常数，

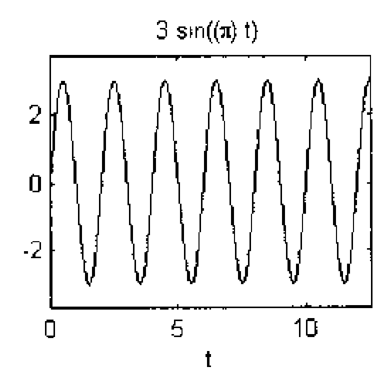
下面我们用 MATLAB 来绘制实指数信号 $f(t) = e^{at}$ 当 $a = -1$、$a = 1$ 及 $a = 0$ 时的时域波形，对应的 MATLAB 命令为：

```
f=sym('exp(a*t)')
```

```
f1=subs(f,'a','-1')
ezplot(f1,[-2,2])
f2=subs(f,'a','1')
ezplot(f2,[-2,2])
f3=subs(f,'a','0')
ezplot(f3,[-2,2])
```

绘制的实指数信号时域波形如图 6-22 所示。





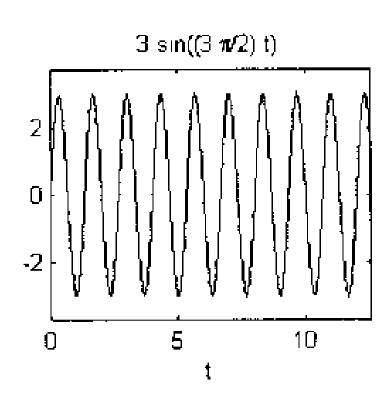


图 6-22 实指数时间信号

从绘制的信号波形我们可以看出：当$a<0$时，实指数信号$f(t)=e^{at}$为随时间衰减的信号；当$a>0$时，$f(t)$为随时间增长的信号；$a=0$时，$f(t)$即是直流信号。

3. 虚指数信号

虚指数信号的一般形式为：

$f(t)=Ae^{j\omega t}$，A为常数，ω为虚指数信号的角频率。

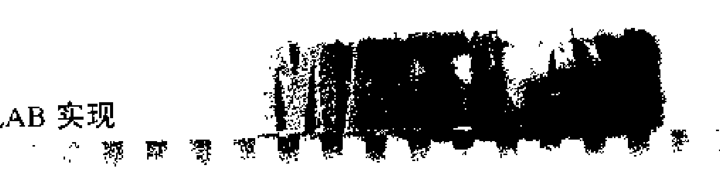
根据欧拉公式有：

$$f(t)=Ae^{j\omega t}=A(\cos\omega t+j\sin\omega t)$$

可见，虚指数信号是时间t的复函数，因此，我们需要用两个实信号来表示虚指数信号，即用模和相角或实部和虚部来表示虚指数信号随时间变化的规律。

下面是用 MATLAB 绘制虚指数信号的实用子函数：

```
function xzsu(w,n1,n2,a)
%n1:绘制波形的起始时间
%n1:绘制波形的终止时间
%w:虚指数信号角频率
%a: 虚指数信号的幅度
t=n1:0.01:n2;
X=a*exp(i*w*t);
Xr=real(X);
Xi=imag(X);
```



```
Xa=abs(X);
Xn=angle(X);
subplot(2,2,1),plot(t,Xr),axis([n1,n2,-(max(Xa)+0.5),max(Xa)+0.5]),
title('实部');
subplot(2,2,3),plot(t,Xi),axis([n1,n2,-(max(Xa)+0.5),max(Xa)+0.5]),
title('虚部');
subplot(2,2,2), plot(t,Xa),axis([n1,n2,0,max(Xa)+1]),title('模');
subplot(2,2,4),plot(t,Xn),axis([n1,n2,-(max(Xn)+1),max(Xn)+1]),title('相角');
```

例 6-10：试用 MATLAB 画出信号 $f(t)=2e^{j\frac{\pi t}{4}}$ 的时域波形（实部、虚部、模及相角），并观察该信号的时域特性。

解：

调用函数 xzsu 即可解决此问题，对应的 MATLAB 调用命令如下：

xzsu(pi/4,0,15,2)

绘制的虚指数信号时域波形如图 6-23 所示。

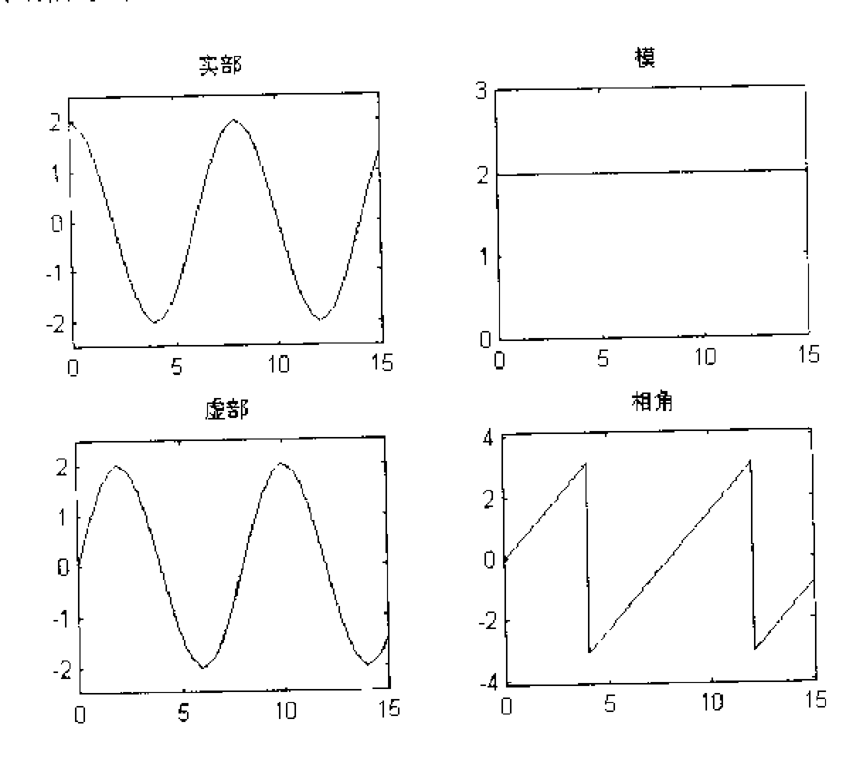


图 6-23　虚指数信号时域波形图

可见虚指数信号 $Ae^{j\omega t}$ 是随时间变化的周期信号，周期 $T=2\pi/\omega$，其实部和虚部分别是等幅的正弦振荡。

4. 复指数信号

复指数信号的基本形式为：

$f(t)=Ae^{st}$，其中 $s=\sigma+j\omega$ 为复常数。由欧拉公式有：

$$f(t)=Ae^{st}=Ae^{\sigma t}e^{j\omega t}=Ae^{\sigma t}\cos(\omega t)+jAe^{\sigma t}\sin(\omega t)$$

式中：实部 $\mathrm{Re}\{f(t)\}=Ae^{\sigma t}\cos(\omega t)$，虚部 $\mathrm{Im}\{f(t)\}=Ae^{\sigma t}\sin(\omega t)$；

可见复指数信号的实部和虚部分别是按指数规律变化的正弦振荡。

当 $\sigma>0$ 时，$f(t)$ 的实部和虚部分别是按指数规律增长的正弦振荡。

当 $\sigma<0$ 时，$f(t)$ 的实部和虚部分别是按指数规律衰减的正弦振荡。

当 $\sigma=0$ 时，$f(t)$ 的实部和虚部分别是等幅的正弦振荡。

例 6-11：画出复指数信号 $f(t)=e^{-t}e^{10it}$ 的实部、虚部、模及相角随时间变化的曲线，并观察其时域特性。

解：

对应的 MATLAB 程序如下：

```
t=0:0.01:3
a=-1;b=10;
z=exp((a+i*b)*t)
subplot(2,2,1),plot(t,real(z)),title('实部')
subplot(2,2,3),plot(t,imag(z)),title('虚部')
subplot(2,2,2),plot(t,abs(z)),title('模')
subplot(2,2,4),plot(t,angle(z)),title('相角')
```

命令执行后，绘制的信号波形如图 6-24 所示。

由绘制的信号波形可以看出，该复指数信号的实部和虚部均为随时间按指数规律衰减的正弦振荡，与理论分析结果完全相符。

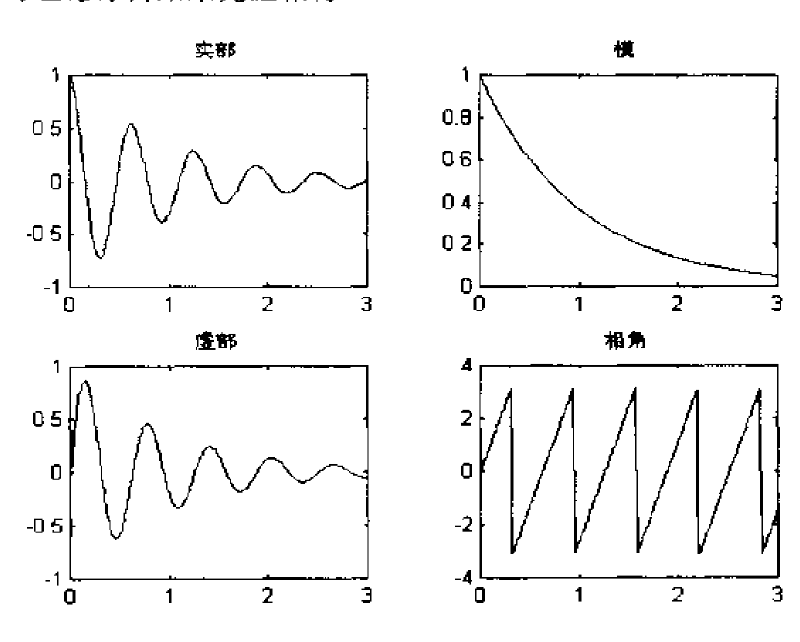


图 6-24　复指数信号时域波形图



6.3.2　离散时间序列

1. 正弦序列

离散时间正弦序列的表达式为：

$$f(k) = A\cos(k\omega + \varphi)$$

式中 k 为无量纲的整数，ω 和 φ 以弧度为单位，ω 称为离散正弦序列的数字角频率，φ 为初相位。

需要注意的是，并非所有的离散时间正弦序列都是周期的。这是因为离散时间信号的自变量和周期序列的周期都必须是整数，在正弦序列中并非对任意 ω 都能找到正整数的周期 N。现在我们就来讨论正弦序列为周期序列的条件。

设 $f(k) = \cos\omega k$ 是周期序列，且周期为 N，则有：

$$\cos\omega k = \cos\omega(k+N) = \cos(\omega k + \omega N)$$

要使上式对所有的 ω 都成立，则 ωN 必须为 2π 的整数倍，即：

$N\omega = 2m\pi$，或 $\dfrac{2\pi}{\omega} = \dfrac{N}{m}$，$m$ 为整数

可见只有当 $\dfrac{2\pi}{\omega}$ 为一有理数时，$\cos\omega k$ 才具有周期性，且周期为 $\dfrac{2\pi m}{\omega}$。

下面我们通过一个实例来说明上述理论结果。

例 6-12：试用 MATLAB 画出正弦序列 $f_1(k) = \cos(k\pi/8)$，$f_2(k) = \cos(2k)$ 的时域波形，观察它们的周期性，并验证是否与理论分析结果相符。

解：

对序列 $f_1(k)$，其角频率 $\omega = \dfrac{\pi}{8}$，$\dfrac{2\pi}{\omega} = 16$，故该序列是周期序列，且周期为 16。

对序列 $f_2(k)$，其角频率 $\omega = 2$，$\dfrac{2\pi}{\omega} = \pi$（无理数），故该序列是非周期序列。

下面我们用 MATLAB 将上述序列的时域波形绘制出来。对应的 MATLAB 命令如下：

```
k=0:40;
subplot(2,1,1)
stem(k,cos(k*pi/8),'filled')
title('cos(k*pi/8)')
subplot(2,1,2)
stem(k,cos(2*k),'filled')
title('cos(2*k)')
```

绘制的序列波形如图 6-25 所示。



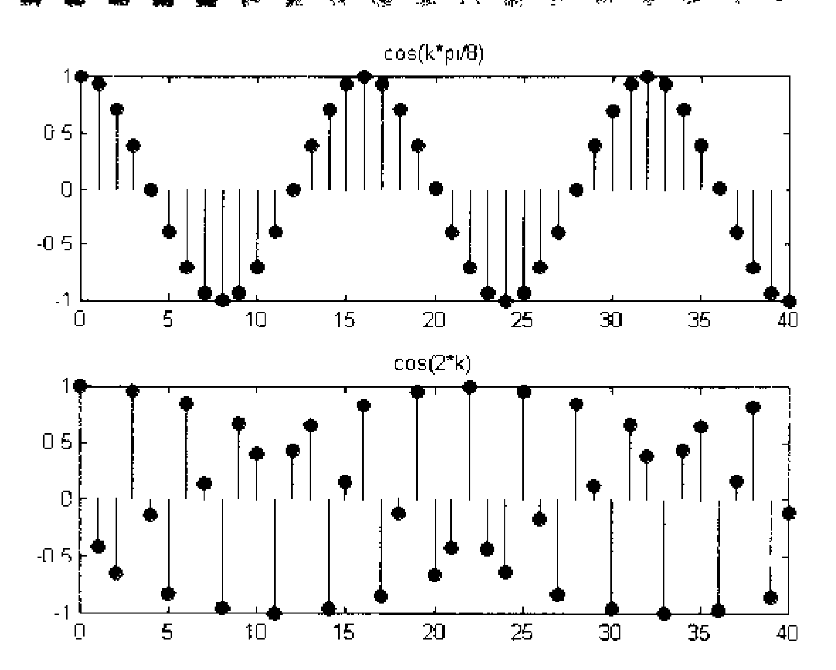


图 6-25 不同频率的正弦序列 subplot(2,1,1),stem(k,cos(k*pi/8),'filled'),title('cos (k*pi/8)')

由绘制的序列波形图，我们可以明显看出，序列 $f_1(k)$ 为周期序列，而 $f_2(k)$ 为非周期序列。

2. 离散时间实指数序列

离散时间实指数序列的一般形式为：

$$f(k)=ca^k，其中 c 和 a 为实常数。$$

我们可以用 MATLAB 编写绘制离散时间实指数序列波形的函数如下：

```
function dszsu(c,a,k1,k2)
%c：指数序列的幅度
%a：指数序列的底数
%k1：绘制序列的起始序号
%k2：绘制序列的终止序号
k=k1:k2;
x=c*(a.^k);
stem(k,x,'filled')
hold on
plot([k1,k2],[0,0])
hold off
```

例 6-13：用 MATLAB 画出序列 $f_1(k)=(\frac{5}{4})^k\varepsilon(k), f_2(k)=(-\frac{3}{4})^k\varepsilon(k)$ 的时域波形，观察两序列的时域特性。

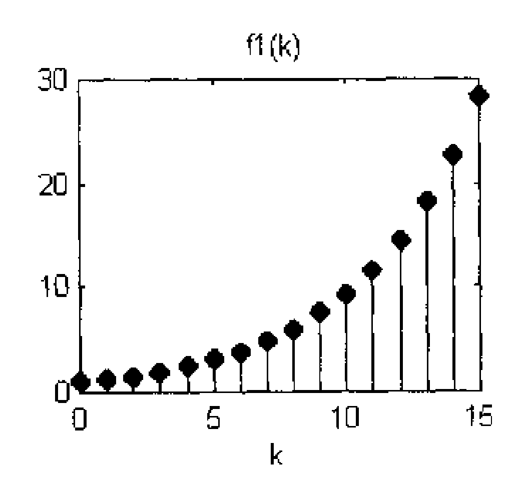
解：



我们可调用前述的 dszsu 函数来解决此问题，对应的 MATLAB 命令为：

```
dszsu(1,5/4,0,40)
xlabel('k')
title('f1(k)')
dszsu(1,-3/4,0,40)
xlabel('k')
title('f2(k)')
```

绘制的离散实指数序列波形如图 6-26 所示。由程序绘制的序列波形图我们可以看出，对离散时间实指数序列 $f(k)=ca^k$，当 a 的绝对值大于 1 时，序列为随时间发散的序列，当 a 的绝对值小于 1 时，序列为随时间收敛的序列。



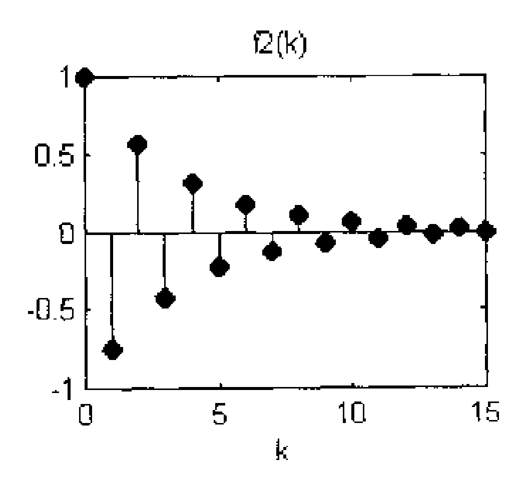


图 6-26　不同底数的实指数序列

3．**离散时间虚指数序列**

离散时间虚指数序列的一般形式为：

$$f(k)=e^{j\omega k}$$

其中 ω 为角频率，根据欧拉公式有：

$$f(k)=e^{j\omega k}=\cos\omega k+j\sin\omega k$$

可见，离散虚指数序列的实部和虚部均是等幅的正弦序列。通过前面的分析可得，虚指数序列的实部和虚部也只有在满足一定的条件时才具有周期性，即当 $2\pi/\omega$ 为有理数时离散虚指数序列才具有周期性(周期 $N=2m\pi/\omega, N/m$ 是最简分数)。

绘制虚指数序列时域波形的 MATLAB 函数如下：

```
function[]=dxzsu(n1,n2,w)
%n1:绘制波形的虚指数序列的起始时间序号
%n2：绘制波形的虚指数序列的终止时间序号
%w：虚指数序列的角频率
k=n1:n2;
```

```
f=exp(i*w*k);
Xr=real(f)
Xi=imag(f)
Xa=abs(f)
Xn=angle(f)
subplot(2,2,1), stem(k,Xr,'filled'),title('实部');
subplot(2,2,3), stem(k,Xi,'filled'),title('虚部');
subplot(2,2,2), stem(k,Xa,'filled'),title('模');
subplot(2,2,4), stem(k,Xn,'filled'),title('相角');
```

例 6-14：用 MATLAB 画出 $f_1(k)=e^{j\frac{k\pi}{4}}$，$f_2(k)=e^{j2k}$ 的时域波形，并分析实部、虚部、相角的周期性与角频率的关系。

解：

实现上述过程的 MATLAB 的命令如下：

```
dxzsu(0,20,pi/4)
dxzsu(0,20,2 )
```

上述命令绘制的虚指数序列波形分别如图 6-27 和图 6-28 所示。由下面两图可见，只有当虚指数序列的角频率满足 $2\pi/\omega$ 为有理数时，信号的实部和虚部和相角都为周期序列，否则为非周期序列。

4. **复指数序列**

复指数序列的一般形式为：

$$f(k)=r^k e^{j\omega k}$$

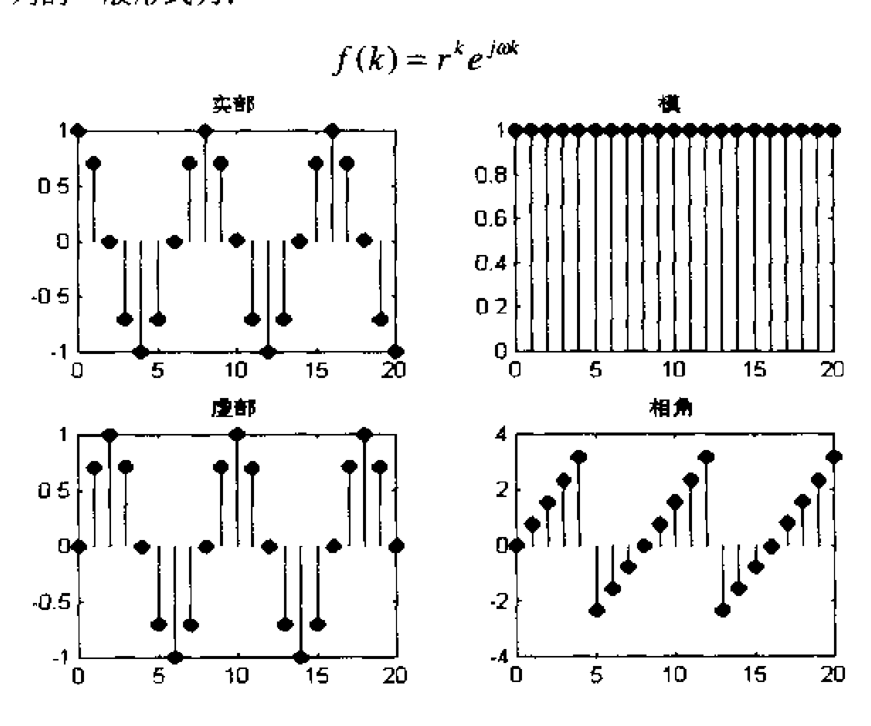


图6-27　周期虚指数序列波形



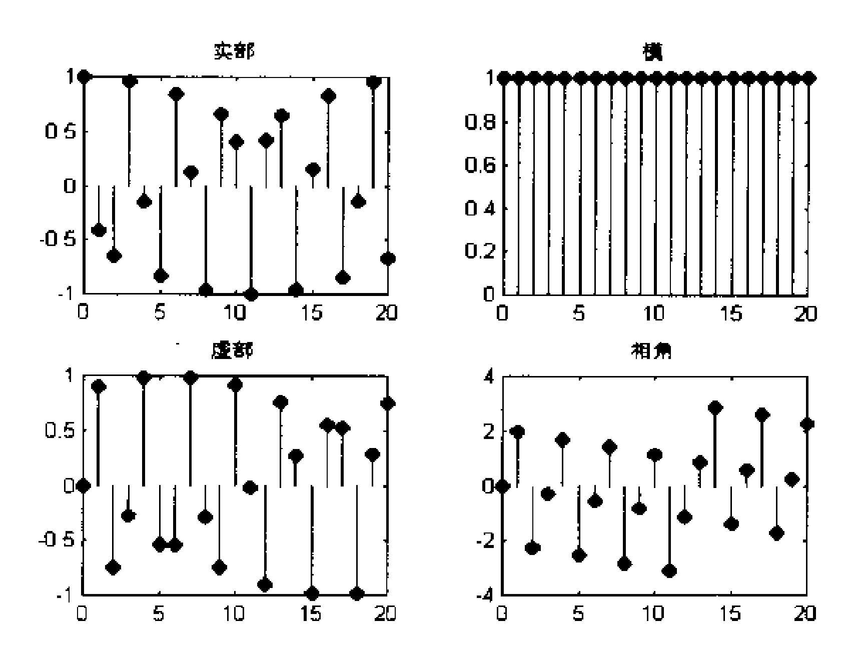


图 6-28　非周期虚指数序列波形

由欧拉公式得：

$$f(k)=r^k e^{j\omega k}=r^k[\cos(\omega k)+j\sin(\omega k)]$$

式中：$\mathrm{Re}\{f(k)\}=r^k\cos(\omega k)$ 为 $f(k)$ 的实部

$\mathrm{Im}\{f(k)\}=r^k\sin(\omega k)$ 为 $f(k)$ 的虚部。

可见，复指数序列 $f(k)$ 的实部和虚部分别为幅度按指数规律变化的正弦序列。

当 $r>1$ 时，$f(k)$ 的实部和虚部分别为指数增长的正弦序列；

当 $0<r<1$ 时，$f(k)$ 的实部和虚部分别为指数衰减的正弦序列；

当 $r=1$ 时，$f(k)$ 的实部和虚部分别为等幅正弦序列。

绘制复指数序列时域波形的 MATLAB 函数如下：

```
function dfzsu(n1,n2,r,w)
%n1:绘制波形的虚指数序列的起始时间序号
%n2：绘制波形的虚指数序列的终止时间序号
%w：虚指数序列的角频率
%r：指数序列的底数
k=n1:n2;
f=(r*exp(i*w)).^k;
Xr=real(f);
Xi=imag(f);
Xa=abs(f);
Xn=angle(f);
subplot(2,2,1), stem(k,Xr,'filled'),title('实部');
```

```
subplot(2,2,3), stem(k,Xi,'filled'),title('虚部');
subplot(2,2,2), stem(k,Xa,'filled'),title('模');
subplot(2,2,4), stem(k,Xn,'filled'),title('相角');
```

例 6-15：用 MATLAB 画出 $f(k)=0.9^k e^{jk\pi/4}$ 的时域波形，观察信号的时域特性与理论结果是否相符。

解：

实现上述过程的 MATLAB 命令如下：

```
dfzsu(0,20,0.9,pi/4)
```

绘制的序列时域波形如图 6-29 所示。

从绘制的序列波形图可以看出，该信号

实部为：$\mathrm{Re}\{f(k)\}=0.9^k\cos(k\pi/4)$

虚部为：$\mathrm{Im}\{f(k)\}=0.9^k\sin(k\pi/4)$

$$r=0.9<1$$

所以，实部和虚部都为指数衰减的正弦序列，其模则为随时衰减的指数序列。

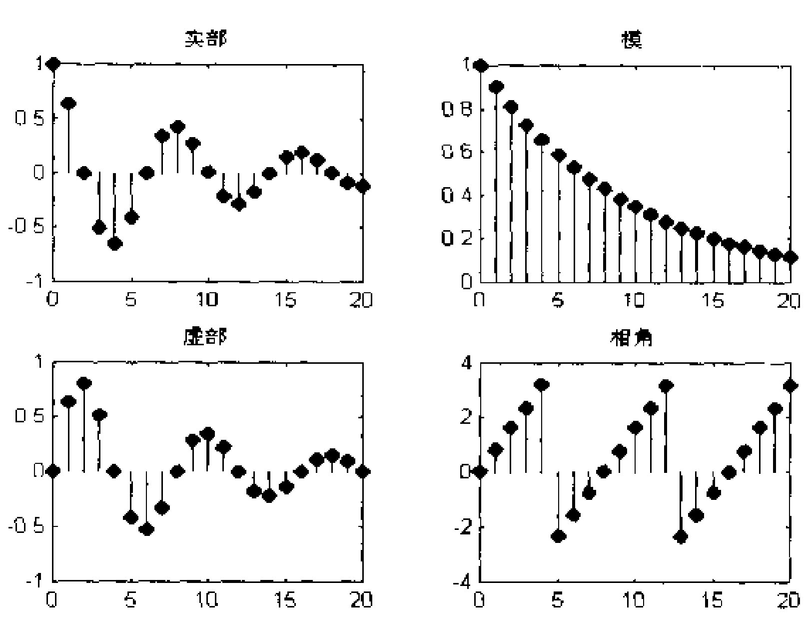


图6-29　复指数序列波形

上机练习题

1．分别用 MATLAB 的向量表示法和符号运算功能，绘出下列信号的时域波形。

（1）$f(t)=(2-e^{-2t})\varepsilon(t)$　　（2）$f(t)=\cos(\pi t)[\varepsilon(t)-\varepsilon(t-1)]$



（3）$f(t)=\varepsilon(-3t+2)$　　（4）$f(t)=-\frac{1}{2}t\varepsilon(t+2)$

2．试用 MATLAB 的向量表示法，绘出下列离散序列的时域波形。

（1）$f(k)=(-\frac{1}{2})^k\varepsilon(k)$　　（2）$f(k)=k[\varepsilon(k)-\varepsilon(k-6)]$

（3）$f(k)=\varepsilon(-k+2)$　　（4）$f(k)=\sin(\frac{k\pi}{4})\varepsilon(k)$

3．试用 MATLAB 绘制出如下连续时间信号的时域波形，并观察信号是否为周期信号，若是周期信号，周期是多少？若不是周期信号，请说明原因。

(1) $f(t)=3\sin(\frac{\pi}{2}t)+2\sin(\pi t)+\sin(2\pi t)$

(2) $f(t)=\sin(t)+\sin(4t)+\sin(5t)$

(3) $f(t)=2e^{j\frac{\pi}{8}t}+e^{j\frac{\pi}{4}t}$

(4) $f(t)=2e^{jt}$

4．试用 MATLAB 绘制双边指数信号 $f(t)=ce^{\alpha|t|}$ 的时域波形，并观察 α 的大小对信号波形的影响。

5．用 MATLAB 绘制出如下离散时间序列的时域波形，并观察信号是否具有周期性？若是周期信号，周期为多少？若不是周期信号，请说明原因。

(1) $f(k)=\cos(\frac{6\pi}{5}k+2)$

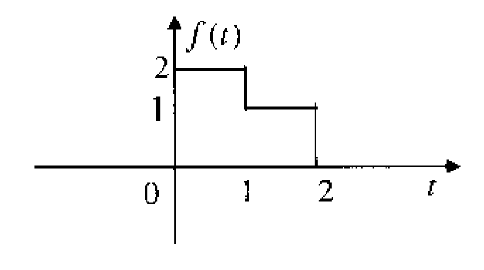
(2) $f(k)=\sin(\frac{1}{4}k)+\sin(\frac{\pi}{4}k)$

(3) $f(k)=\sin(\frac{\pi}{4}k)+\cos(\frac{\pi}{8}k)+\sin(\frac{\pi}{2}k)$

(4) $f(k)=2e^{j2\pi k}$

(5) $f(k)=2e^{j\frac{\pi}{8}k}+e^{j2k}$

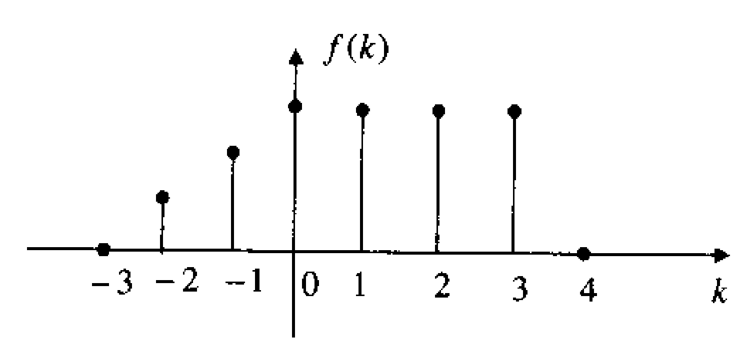
6．已知信号波形如下图所示，试用 MATLAB 绘出满足下列要求的信号波形。



(1) $f(-t)$　　(2) $f(t-2)$

(3) $f(1-2t)$　　(4) $f(\frac{1}{2}t+1)$

7. 已知离散序列波形如下图所示，试用 MATLAB 绘出满足下列要求的序列波形。



(1) $f(k-2)\varepsilon(k)$　　(2) $f(-k)$

(3) $f(-k+2)$　　(4) $f(k-2)\varepsilon(k-2)$

第 7 章　线性系统的时域分析及 MATLAB 实现



7.1 离散时间序列卷积和 MATLAB 实现

离散时间序列 $f_1(k)$ 和 $f_2(k)$ 的卷积和定义为

$$f(k)=f_1(k)*f_2(k)=\sum_{i=-\infty}^{\infty}f_1(i)\cdot f_2(k-i) \tag{7-1}$$

在离散信号与系统的分析过程中，我们有两个与卷积和相关的重要结论，这就是：

（1）$f(k)=\sum_{i=-\infty}^{\infty}f(i)\cdot\delta(k-i)=f(k)*\delta(k)$，即离散序列可分解为一系列幅度由 $f(k)$ 决定的单位序列 $\delta(k)$ 及其平移序列之和。

（2）对线性时不变系统，设其输入序列为 $f(k)$，单位响应为 $h(k)$，其零状态响应为 $y(k)$，则有：$y(k)=\sum_{i=-\infty}^{\infty}f(i)\cdot h(k-i)=f(k)*h(k)$

可见，离散序列卷积和的计算对进行离散信号与系统的分析具有非常重要的意义。

设序列 $f_1(k)$ 在区间 $n_1\sim n_2$ 非零，$f_2(k)$ 在区间 $m_1\sim m_2$ 非零，则 $f_1(k)$ 的时域宽度为 $L_1=n_2-n_1+1$，$f_2(k)$ 的时域宽度为 $L_2=m_2-m_1+1$。由卷积和的定义可得，序列 $f(k)=f_1(k)*f_2(k)$ 的时域宽度为 $L=L_1+L_2-1$，且只在区间（n_1+m_2）$\sim n_1+m_2+(L_1+L_2)-2$ 非零。因此，对于 $f_1(k)$ 和 $f_2(k)$ 均为有限期间非零的情况，我们只需要计算序列 $f(k)$ 在区间 $n_1+m_2\sim n_1+m_2+(L_1+L_2)-2$ 的序列值，便可以表征整个序列 $f(k)$。

MATLAB 的 conv()函数可以帮助我们快速求出两个离散序列的卷积和。conv 函数的调用格式为：

```
f = conv (f1, f2 )
```

其中 $f1$ 为包含序列 $f_1(k)$ 的非零样值点的行向量，$f2$ 为包含序列 $f_2(k)$ 的非零样值点的行向量，向量 f 则返回序列 $f(k)=f_1(k)*f_2(k)$ 的所有非零样值点行向量。

例如，已知序列 $f_1(k)$ 和 $f_2(k)$ 如下所示：

$$f_1(k)=\begin{cases}1 & 0\leqslant k\leqslant 2\\ 0 & \text{其他}\end{cases}\qquad f_2(k)=\begin{cases}1 & k=1\\ 2 & k=2\\ 3 & k=3\\ 0 & \text{其他}\end{cases}$$

则调用 conv()函数求上述两序列的卷积和的 MATLAB 命令为：

```
f1=ones(1,3);
f2=0:3;
```

```
f=conv(f1,f2)
```

运行结果为：

```
f =
     0     1     3     6     5     3
```

由上例可以看出，函数 conv()不需要给定序列 $f_1(k)$ 和 $f_2(k)$ 非零样值点的时间序号，也不返回序列 $f(k)=f_1(k)*f_2(k)$ 的非零样值点的时间序号。因此，要正确地标识出函数 conv()的计算结果向量 f，我们还必须构造序列 $f_1(k)$、$f_2(k)$ 及 $f(k)$ 的对应序号向量。对于上例，设序列 $f_1(k)$、$f_2(k)$ 及 $f(k)$ 的对应序号向量分别为 $k1$、$k2$ 和 k，则应有：

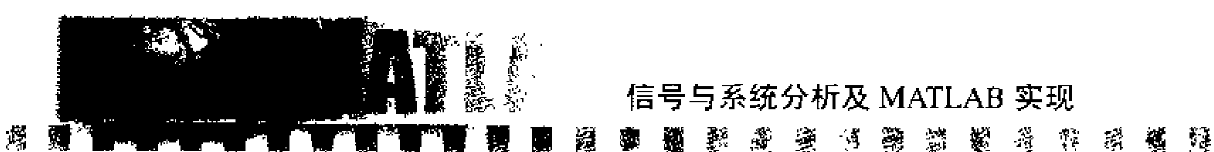
```
k1=[0, 1, 2 ];
k2=[0, 1, 2, 3];
k=[0, 1, 2, 3, 4, 5];
```

如前所述，$f(k)$ 的序号向量 k 由序列 $f_1(k)$ 和 $f_2(k)$ 的非零样值点的起始序号及它们的时域宽度决定。故上例最终的卷积和结果应为：

$$f(k)=\begin{cases} 0 & k=0 \\ 1 & k=1 \\ 3 & k=2 \\ 6 & k=3 \\ 5 & k=4 \\ 3 & k=5 \\ 0 & \text{其他} \end{cases}$$

下面是利用 MATLAB 计算两离散序列卷积和 $f(k)=f_1(k)*f_2(k)$ 的实用函数 dconv( )，该程序在计算出卷积和 $f(k)$ 的同时，还绘出序列 $f_1(k)$、$f_2(k)$ 和 $f(k)$ 的时域波形图，并返回 $f(k)$ 的非零样值点的对应向量。

```
function [f,k]=dconv(f1,f2,k1,k2)
%The function of compute   f=f1*f2
% f:   卷积和序列 f(k)对应的非零样值向量
% k:    序列 f(k)的对应序号向量
% f1: 序列 f1(k)非零样值向量
% f2:   序列 f2(k)的非零样值向量
% k1:   序列 f1(k)的对应序号向量
% k2:   序列 f2(k)的对应序号向量
f=conv(f1,f2)                          %计算序列 f1 与 f2 的卷积和 f
k0=k1(1)+k2(1);                        %计算序列 f 非零样值的起点位置
k3=length(f1)+length(f2)-2;            %计算卷积和 f 的非零样值的宽度
k=k0:k0+k3                             %确定卷积和 f 非零样值的序号向量
subplot(2,2,1)
stem(k1,f1)                            %在子图 1 绘序列 f1(k)时域波形图
title('f1(k)')
xlabel('k')
```




```
ylabel('f1(k)')
subplot(2,2,2)
stem(k2,f2)                                    %在图 2 绘序列 f2(k)时波形图
title('f1(k)')
xlabel('k')
ylabel('f2(k)')
subplot(2,2,3)
stem(k,f);                                     %在子图 3 绘序列 f(k)的波形图
title('f(k)f1(k)与 f2(k)的卷积和 f(k)')
xlabel('k')
ylabel('f(k)')
h=get(gca,'position');
h(3)=2.5*h(3);
set(gca,'position',h)                          %将第三个子图的横坐标范围扩为原来的 2.5 倍
```

例 7-1：试用 MATLAB 计算如下所示序列 $f_1(k)$ 与 $f_2(k)$ 的卷积和 $f(k)$，绘出它们的时域波形，并说明序列 $f_1(k)$ 和 $f_2(k)$ 的时域宽度与序列 $f(k)$ 的时域宽度的关系。

$$f_1(k)=\begin{cases}1 & k=-1\\ 2 & k=0\\ 1 & k=1\\ 0 & \text{其他}\end{cases}\qquad f_2(k)=\begin{cases}1 & -2\leqslant k\leqslant 2\\ 0 & \text{其他}\end{cases}$$

解：

该问题可用上述介绍的 dconv( )函数来解决，实现这一过程的 MATLAB 命令如下：

```
f1=[1 2 1];
k1=[-1 0 1];
f2=ones(1,5);
k2=-2:2;
[f, k]=dconv(f1,f2,k1,k2)
```

运行结果为：

```
f =
     1     3     4     4     4     3     1
k =
-3    -2    -1     0     1     2     3
```

程序绘制的序列时域波形图如图 7-1 所示。



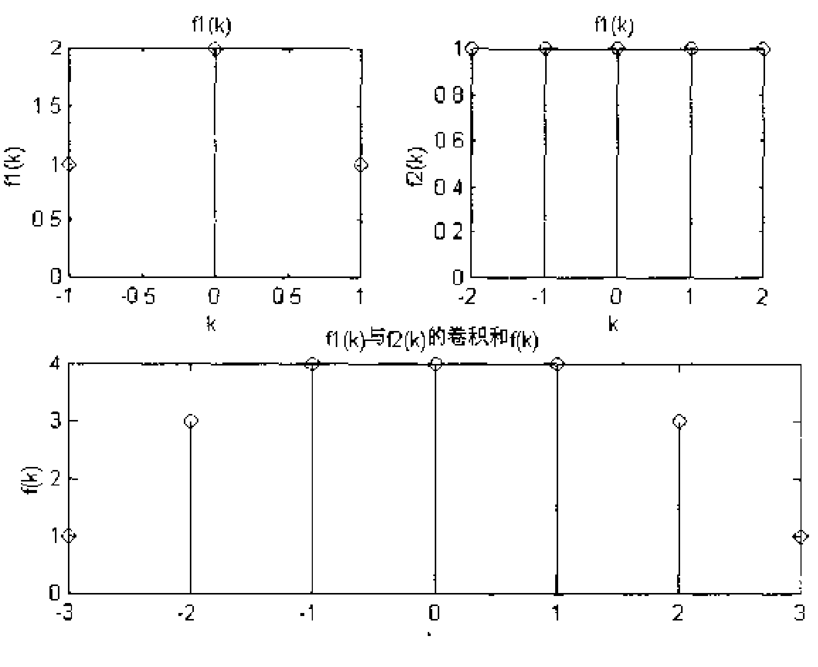



图 7-1　离散序列卷积和 1

由程序运行结果及绘制的波形可以看出，序列 $f(k)$ 的时域宽度等于序列 $f_1(k)$ 和 $f_2(k)$ 的时域宽度之和减一。

例 7-2：已知某 LTI 离散系统，其单位响应 $h(k)=\varepsilon(k)-\varepsilon(k-4)$，求该系统在激励为 $f(k)=\varepsilon(k)-\varepsilon(k-3)$ 时的零状态响应 $y(k)$，并绘出其时域波形图。

解：

由 LTI 变离散系统的分析可得：

$y(k)=h(k)*f(k)$

因此，我们可调用 dconv()函数来解决此问题，相应的 MATLAB 命令为：

```
f1=ones(1,4);
k1=0:3;
f2=ones(1,3);
k2=0:2;
[f,k]=dconv(f1,f2,k1,k2)
```

运行结果为：

```
f =
     1     2     3     3     2     1
k =
0     1     2     3     4     5
```

程序绘制的序列时域波形图如图 7-2 所示。

图 7-2 离散序列卷积 2

需要注意的是，调用 conv()函数计算序列卷积和时，该函数将向量 *f*1 和 *f*2 以外的序列样值均视为零，因此，当序列 $f_1(k)$ 或 $f_2(k)$ 为无限长序列时调用 conv()函数就可能会产生误差。例如，考虑如下序列：

$$f_1(k)=\varepsilon(k) \quad f_2(k)=\varepsilon(k)-\varepsilon(k-3)$$

则用 conv 函数来计算卷积和会出现什么情况呢？由于 $f_1(k)$ 为时间无限长序列，因此在定义向量 *f*1 时，必须将 $f_1(k)$ 进行截断，如定义：

```
f1=ones(1,100);
f2=ones(1,3);
```

这时，调用函数 conv（*f*1,*f*2）计算卷积和时，该函数便将向量 *f*1 对应的序列 $f_1(k)$ 在区间 0~100 以外均视为零，故此时，conv（*f*1,*f*2）计算出的卷积和样值点只有部分是真实的。

7.2 连续时间信号卷积及 MATLAB 实现

本节向读者介绍连续时间信号卷积及 MATLAB 实现。

7.2.1 卷积积分

卷积积分在信号与线性系统分析中具有非常重要的意义，是信号与系统分析的基本方法之一。下面我们简单回顾下卷积积分的基本方法和原理。

设 $p_\Delta(t)$ 为时域宽度为 Δ，高度为 $\dfrac{1}{\Delta}$ 的矩形门信号，如图 7-3 所示。



让信号 $p_{\Delta}(t)$ 通过某 LTI 连续系统，产生的响应为 $h_{\Delta}(t)$，如图 7-3 所示。

根据冲激信号的定义有：

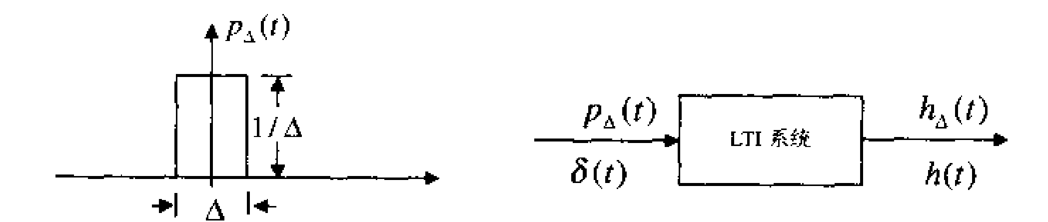


图 7-3　$h_{\Delta}(t)$ 响应

$$\delta(t)=\lim_{\Delta\to 0}p_{\Delta}(t)$$

显然，该系统的冲激响应为：

$$h(t)=\lim_{\Delta\to 0}h_{\Delta}(t)$$

现考虑连续信号 $f(t)$，我们可对该信号进行时域分解，即将 $f(t)$ 近似表示成一系列信号 $p_{\Delta}(t)$ 的时间平移信号的线性组合，即：

$$f(t)=f_{\Delta}(t)=\sum_{k=-\infty}^{\infty}f(k\Delta)\cdot p_{\Delta}(t-k\Delta)\cdot\Delta$$

将 $f_{\Delta}(t)$作为激励信号接入上述 LTI 系统，则由线性系统的性质可得，此时系统零状态响应

$$y_{\Delta}(t)=\sum_{k=-\infty}^{\infty}f(k\Delta)\cdot h_{\Delta}(t-k\Delta)\cdot\Delta$$

当 $\Delta\to 0$ 时，显然有：

$$f(t)=\lim_{\Delta\to 0}f_{\Delta}(t)=\lim_{\Delta\to 0}\sum_{k=-\infty}^{\infty}f(k\Delta)\cdot p_{\Delta}(t-k\Delta)\cdot\Delta=\int_{-\infty}^{\infty}f(\tau)\cdot\delta(t-\tau)\mathrm{d}\pi \qquad (7\text{-}2)$$

$$y(t)=\lim_{\Delta\to 0}y_{\Delta}(t)=\lim_{\Delta\to 0}\sum_{k=-\infty}^{\infty}f(k\Delta)\cdot h_{\Delta}(t-k\Delta)\cdot\Delta=\int_{-\infty}^{\infty}f(\tau)\cdot h(t-\tau)\mathrm{d}\pi \qquad (7\text{-}3)$$

式（7-2）和（7-3）在数学上具有相同的规律，因此我们将连续时间信号 $f_1(t)$ 和 $f_2(t)$ 的卷积积分（简称为卷积）$f(t)$定义为：

$$f(t)=f_1(t)*f_2(t)=\int_{-\infty}^{\infty}f_1(\tau)\cdot f_2(t-\tau)\mathrm{d}\tau$$

从以上的连续信号与系统的分析过程，我们可得出两个与卷积相关的重要结论，这就

是：

（1） $f(t) = f(t) * \delta(t)$，即连续信号可分解为一系列幅度由 $f(t)$ 决定的冲激信号 $\delta(t)$ 及其平移信号之和。

（2）线性时不变连续系统，设其输入信号为 $f(t)$，单位响应为 $h(t)$，其零状态响应为 $y(t)$，则有：$y(t) = f(k) * h(k)$。

可见，连续信号卷积的计算对我们进行连续信号与系统的分析具有非常重要的意义。

7.2.2 用 MATLAB 实现连续时间信号的卷积

由上面的分析我们可以看出，卷积积分运算实际上可用信号的分段求和来实现，即：

$$f(t) = f_1(t) * f_2(t) = \int_{-\infty}^{\infty} f_1(\tau) \cdot f_2(t-\tau) d\tau = \lim_{\Delta \to 0} \sum_{k=-\infty}^{\infty} f_1(k\Delta) \cdot f_2(t - k\Delta) \cdot \Delta$$

如果我们只求当 $t = n\Delta$（n 为整数）时 $f(t)$ 的值 $f(n\Delta)$，则由上式可得：

$$f(n\Delta) = \sum_{k=-\infty}^{\infty} f_1(k\Delta) \cdot f_2(n\Delta - k\Delta) \cdot \Delta = \Delta \sum_{k=-\infty}^{\infty} f_1(k\Delta) \cdot f_2[(n-k)\Delta] \qquad (7\text{-}4)$$

式（7-4）中的 $\sum_{k=-\infty}^{\infty} f_1(k\Delta) \cdot f_2[(n-k)\Delta]$ 实际上就是连续信号 $f_1(t)$ 和 $f_2(t)$ 经等时间隔 Δ 均匀抽样的离散序列 $f_1(k\Delta)$ 和 $f_2(k\Delta)$ 的卷积和。当 Δ 足够小时，$f(n\Delta)$ 就是卷积积分的结果——连续时间信号 $f(t)$ 的较好的数值近似。

因此，用 MATLAB 实现连续信号 $f_1(t)$ 与 $f_2(t)$ 卷积的过程如下：

将连续信号 $f_1(t)$ 与 $f_2(t)$ 以时间隔 Δ 进行取样，得到离散序列 $f_1(k\Delta)$ 和 $f_2(k\Delta)$；

构造与 $f_1(k\Delta)$ 和 $f_2(k\Delta)$ 相对应的时间向量 $k1$ 和 $k2$（注意，此时时间序号向量 k1 和 k2 的元素不再是整数，而是取样时间隔 Δ 的整数倍的时间间隔点）；

调用 conv()函数计算卷积积分 $f(t)$ 的近似向量 $f(n\Delta)$；

构造 $f(n\Delta)$ 对应的时间向量 k。

可见，我们只要对前面介绍的计算离散序列卷积和及绘制序列波形的子程序 dconv() 进行适当的改变，即可编写出计算连续时间信号卷积积分的数值近似并绘制其时域波形的通用函数。

下面即是利用 MATLAB 实现连续信号卷积的通用函数 sconv( )，该程序在计算出卷积积分的数值近似的同时，还绘出 $f(t)$ 的时域波形图。需要注意的是，程序中是如何构造 $f(t)$ 的对应时间向量 k 的？另外，程序在绘制 $f(t)$ 波形图时采用的是 plot 命令而不是 stem 命令。

```
function   [f,k]=sconv(f1,f2,k1,k2,p)
```

```
%计算连续信号卷积积分 f(t)=f1(t)*f2(t)
% f:   卷积积分 f(t)对应的非零样值向量
% k:   f(t)的对应时间向量
% f1: f1(t) 非零样值向量
% f2: f2(t)的非零样值向量
% k1: f1(t)的对应时间向量
% k2:  序列 f2(t)的对应时间向量
% p:    取样时间间隔
f=conv(f1,f2);                          %计算序列 f1 与 f2 的卷积和 f
f=f*p;
k0=k1(1)+k2(1);                         %计算序列 f 非零样值的起点位置
k3=length(f1)+length(f2)-2;             %计算卷积和 f 的非零样值的宽度
k=k0:p:k3*p;                            %确定卷积和 f 非零样值的时间向量
subplot(2,2,1)
plot(k1,f1)                             %在子图 1 绘 f1(t)时域波形图
title('f1(t)')
xlabel('t')
ylabel('f1(t)')
subplot(2,2,2)
plot(k2,f2)                             %在子图 2 绘 f2(t)时波形图
title('f2(t)')
xlabel('t')
ylabel('f2(t)')
subplot(2,2,3)
plot(k,f);                              %画卷积 f(t)的时域波形
h=get(gca,'position');
h(3)=2.5*h(3);
set(gca,'position',h)                   %将第三个子图的横坐标范围扩为原来的 2.5 倍
title('f(t)=f1(t)*f2(t)')
xlabel('t')
ylabel('f(t)')
```

下面我们举例来说明，如何利用上述子程序来求连续时间信号的卷积。

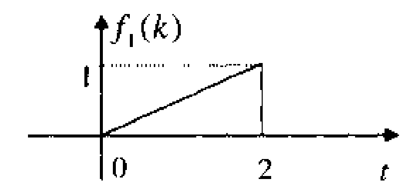
例 7-3：已知两连续时间信号如下图所示，试用 MATLAB 求 $f(t)=f_1(t)*f_2(t)$，并绘出 $f(t)$ 的时域波形图。

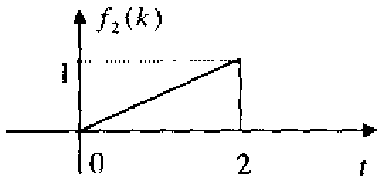
解：

我们可以调用前述的函数 sconv( )来解决此问题，即首先设定取样时间间隔 p，并对连续信号 $f_1(t)$ 和 $f_2(t)$ 的非零值区间以时间间隔 p 进行抽样，产生离散序列 $f1$ 和 $f2$，然后构造离散序列 $f1$ 和 $f2$ 所对应的时间向量 $k1$ 和 $k2$，最后再调用 sconv 函数即可求出 $f_1(t)*f_2(t)$ 的数值近似，并绘出其时域波形图。

实现上述过程的 MATLAB 命令如下：

```
p=0.01;
k1=0:p:2;
f1=0.5*k1;
k2=k1;
f2=f1;
[f,k]=sconv(f1,f2,k1,k2,p)
```





上述命令绘制的波形图如图 7-4 所示。图 7-4 中分别给出了取样时间间隔 p=0.5 和 p=0.01 时的处理效果。可见，当取样时间 p 足够小时，函数 sconv( )的计算结果就是连续时间卷积 $f(t)=f_1(t)*f_2(t)$ 的较好的数值近似。

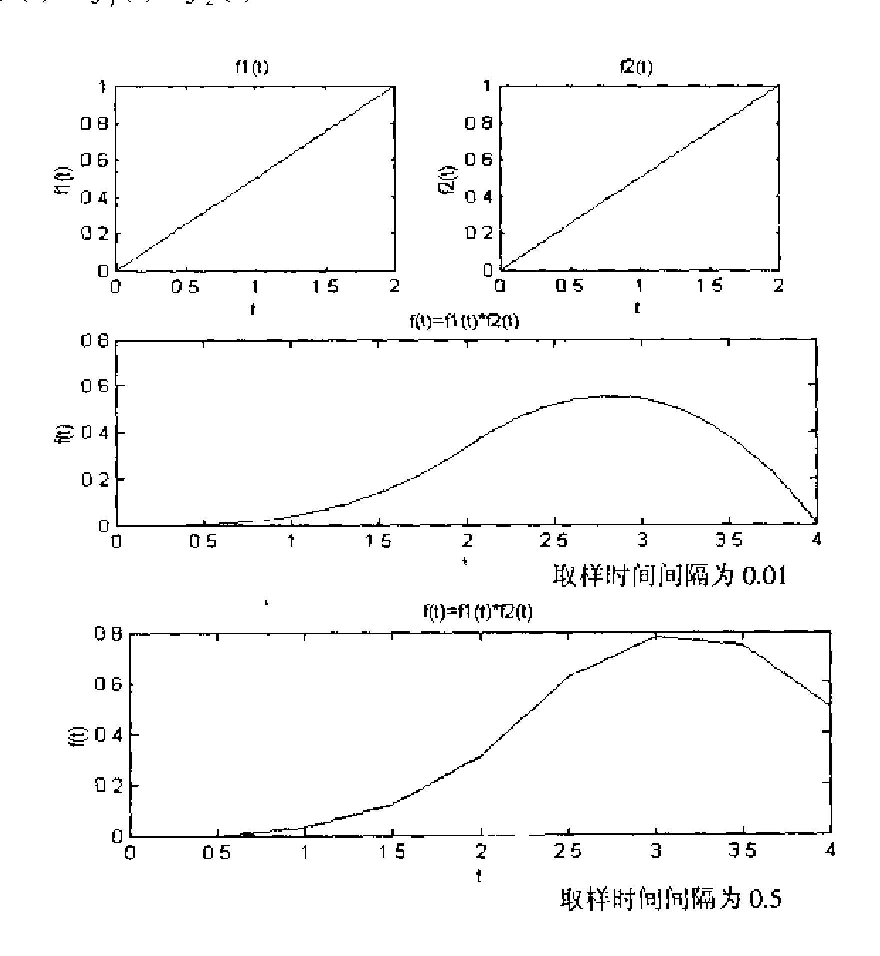
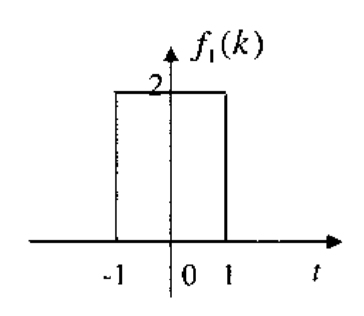


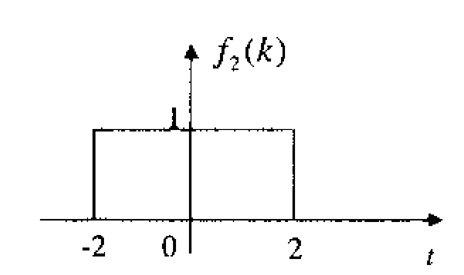
图 7-4　例 7-3 的连续信号卷积和 1

例 7-4：已知两连续时间信号如下图所示，试用 MATLAB 求 $f(t)=f_1(t)*f_2(t)$，




并绘出 $f(t)$ 的时域波形图。





解：

此题和上例基本相同，我们可以用与上例相同的方法调用函数 sconv( )函数来解决此问题。所不同的是，要重新定义连续信号 $f_1(t)$ 和 $f_2(t)$ 的非零值区间以时间间隔 p 进行抽样产生离散序列 f1 和 f2，以及 $f_1(t)$ 和 $f_2(t)$ 相对应的时间向量 k1 和 k2。实现上述过程的 MATLAB 命令如下：

```
p=0.01;
k1=-1:p:1
f1=2*ones(1,length(k1))
k2=-2:p:2
f2=ones(1,length(k2))
[f,k]=sconv(f1,f2,k1,k2,p)
```

绘制的卷积结果波形如图 7-5 所示。

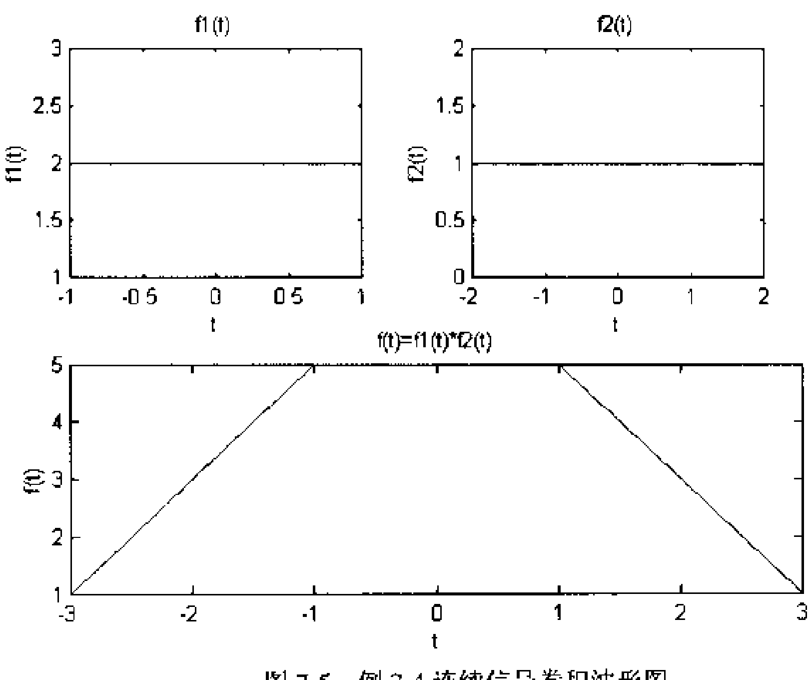


图 7-5 例 7-4 连续信号卷积波形图



7.3 连续系统的冲激响应、阶跃响应及 MATLAB 实现

LTI 系统当输入为冲激信号 $\delta(t)$ 时产生的零状态响应称为系统的冲激响应，用 $h(t)$ 表示。若输入为单位阶跃信号 $\varepsilon(t)$ 时系统产生的零状态响应则称为系统的阶跃响应，记为 $g(t)$，如图 7-6 所示。

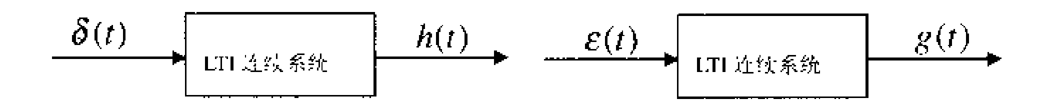


图 7-6 冲激和阶跃响应示

对 LTI 连续系统，设其输入信号为 $f(t)$，冲激响应为 $h(t)$，零状态响应为 $y(t)$，则有：

$$y(t)=f(t)*h(t)$$

即 $h(t)$ 包含了连续系统的固有特性，与系统的输入无关。我们只要知道了系统的冲激响应，即可求得系统在不同输入时产生的输出。

因此，求解系统的冲激响应 $h(t)$ 对我们进行连续系统的分析具有非常重要的意义。

MATLAB 为用户提供了专门用于求连续系统冲激响应及阶跃响应，并绘制其时域波形的函数 impulse 和 step。

在调用函数 impulse()和 step()时，我们需要用向量来对连续系统进行表示。

设描述连续系统的微分方程为：

$$\sum_{i=0}^{N}a_i y^{(i)}(t)=\sum_{j=0}^{M}b_j f^{(j)}(t)$$

则我们可以用向量 a 和 b 来表示该系统，即：

$$a=[a_N,a_{N-1},\cdots\cdots,a_1,a_0]$$

$$b=[b_M,b_{M-1},\cdots\cdots,b_1,b_0]$$

注意，在用向量来表示微分方程描述的连续系统时，向量 a 和 b 的元素一定要以微分方程时间求导的降幂次序来排列，且缺项要用 0 来补齐。例如对微分方程

$$2y''(t)+3y'(t)+6y(t)=f(t)$$

则表示该系统的对应向量应为：

a=[2 3 6];




b=[1];

而对微分方程：

$$y''(t)+3y'(t)+2y(t)=f''(t)+f(t)$$

则表示该系统的对应向量应为：

a=[1 3 2];

b=[1 0 1];

1．impulse()函数

函数 impulse()将绘出由向量 a 和 b 表示的连续系统在指定时间范围内的冲激响应 $h(t)$ 的时域波形图，并能求出指定时间范围内冲激响应的数值解。Impulse()函数有如下几种调用格式：

- impulse(b, a)

该调用格式以默认方式绘出向量 a 和 b 定义的连续系统的冲激响应的时域波形。例如，若描述某连续系统的微分方程为：

$$y''(t)+5y'(t)+6y(t)=3f''(t)+2f(t)$$

运行如下 MATLAB 命令：

```
a=[1 5 6];
b=[3 2];
impulse(b,a)
```

则绘出系统的冲激响应波形，如图 7-7 所示。

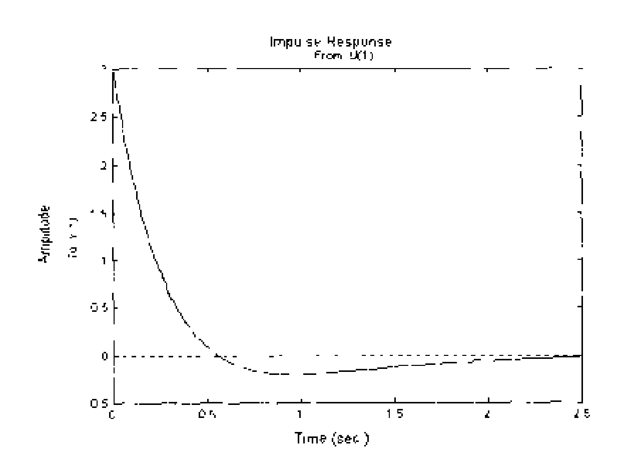


图 7-7　连续系统的冲激响应 1

- impulse(b, a, t)

该调用格式将绘出由向量 a 和 b 定义的连续系统在 0~t 时间范围内冲激响应的时域波形。对上例，若运行如下命令：

```
impulse(b,a,10)
```

则绘出系统在 0~10 秒范围内冲激响应的时域波形，如图 7-8 所示。

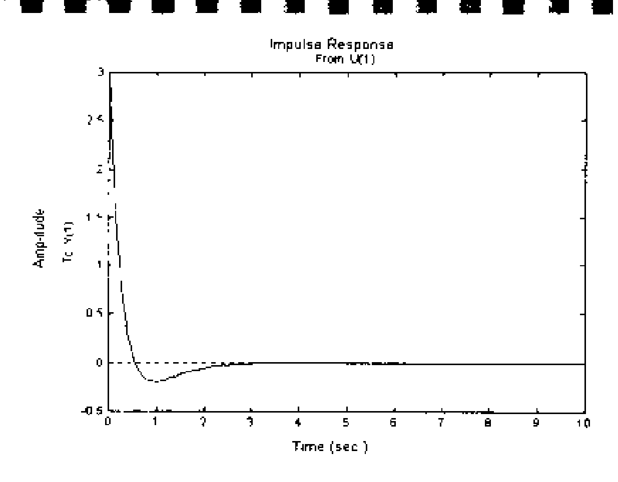


图 7-8 连续系统的冲激响应 2

- impulse(b, a, t1:p:t2)

该调用格式将绘出由向量 *a* 和 *b* 定义的连续系统在 *t*1~*t*2 时间范围内，且以时间间隔 p 均匀取样的冲激响应的时域波形。对上例，若运行命令：

```
impulse(b,a,1:0.1:2)
```

则绘出系统在 1~2 秒范围内，并以时间间隔 0.1 秒取样的冲激响应的时域波形，如图 7-9 所示。

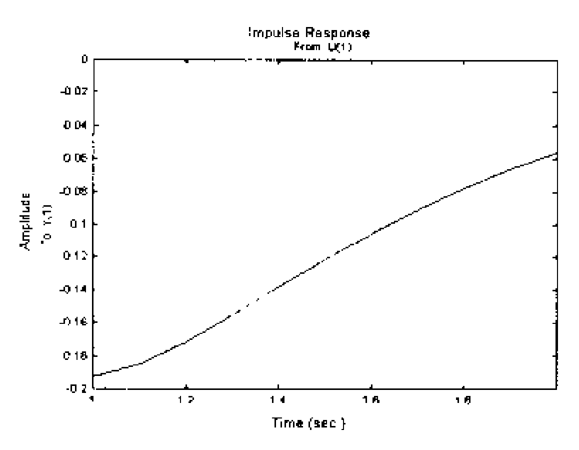


图 7-9 连续系统的冲激响应 3

- y=impulse(b, a, t1:p:t2)

该调用格式并不绘出系统冲激响应的波形，而是求出向量 *a* 和 *b* 定义的连续系统在 *t*1~*t*2 时间范围内以时间间隔 p 取样的系统冲激响应的数值解。对上例，若运行命令：

```
y=impulse(b,a,0:0.2:2)
```

则运行结果为：

```
y =
    3.0000
    1.1604
    0.3110
   -0.0477
```

```
-0.1726
-0.1928
-0.1716
-0.1383
-0.1054
-0.0777
-0.0559
```

2．step()函数

函数 step()将绘出由向量 a 和 b 表示的连续系统的阶跃响应 $g(t)$ 在指定时间范围内的波形图，并能求出其数值解。和 impulse()函数一样，step()函数也有如下四种调用格式：

```
step(b,a)
step(b,a,t)
step(b,a,t1:p:t2)
y=step(b,a,t1:p:t2)
```

上述调用格式的功能和 impulse()函数完全相同，所不同的是命令绘制的是系统的阶跃响应 $g(t)$ 的曲线而不冲激响应 $h(t)$ 的曲线。对上例，若执行命令：

```
step(b,a)
```

则绘制的系统阶跃响应时域波形如图 7-10 所示。

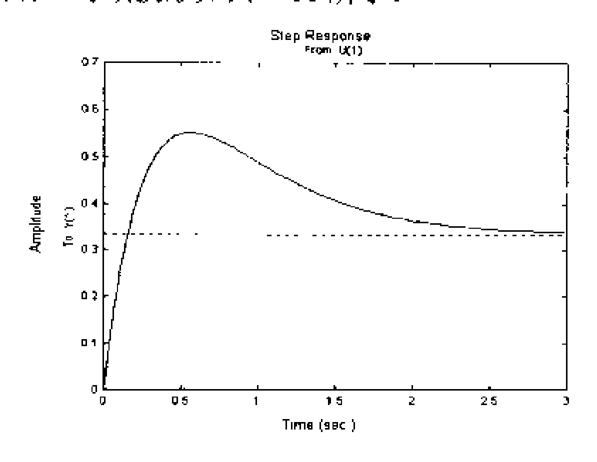


图 7-10　连续系统的阶跃响应

例 7-5：已知描述某连续系统的微分方程为：

$$2y''(t)+y'(t)+8y(t)=f(t)$$

试用 MATLAB 绘出该系统的冲激响应和阶跃响应的波形。

解：

直接调用函数 impulse()和 step()即可解决此问题，对应的 MATLAB 命令如下：

```
b=[1];
a=[2 1 8];
subplot(1,2,1)
```

```
impulse(b,a)
subplot(1,2,2)
step(b,a)
```

上述命令绘制的系统冲激响应和阶跃响应的波形如图 7-11 所示。

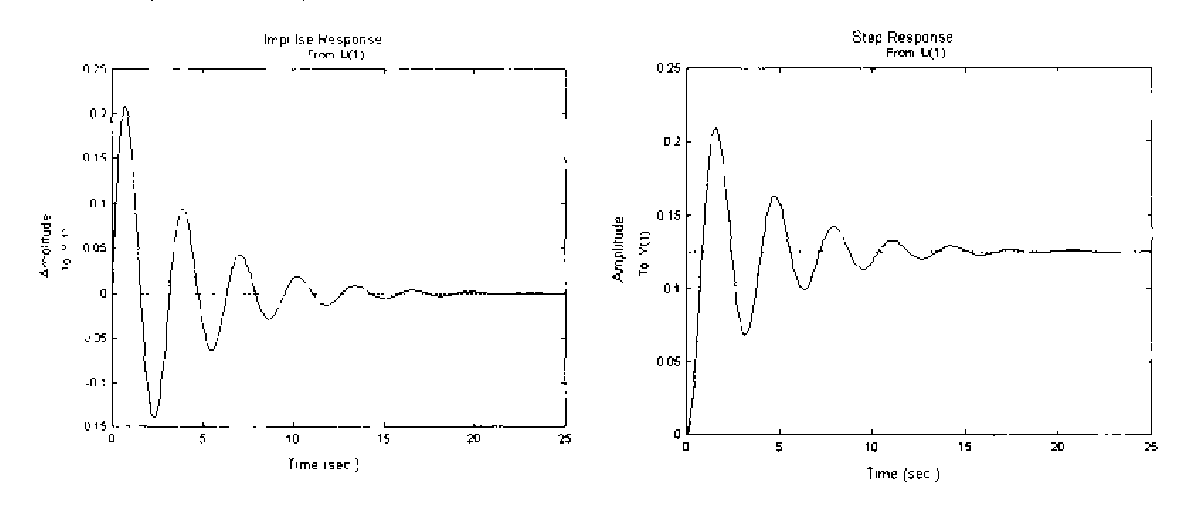


图 7-11 例 7-5 系统冲激和阶跃响应

7.4 离散系统的单位响应及 MATLAB 实现

LTI 离散系统当输入为单位序列$\delta(k)$时产生的零状态响应称为系统的单位响应，用$h(k)$表示。系统输入为单位阶跃序列$\varepsilon(k)$时产生的零状态响应称为系统的单位阶跃响应，记为$g(k)$，如图 7-12 所示。

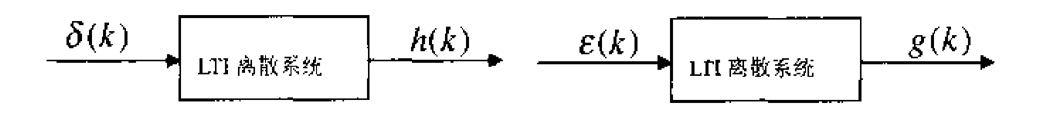


图 7-12 单位响应和阶跃响应示意图

对 LTI 离散系统，设其输入序列为$f(k)$，单位响应为$h(k)$，零状态响应为$y(k)$，则有：

$$y(k)=f(k)*h(k)$$

即$h(k)$包含了离散系统的固有特性，与输入序列无关。我们只要知道了离散系统的单位响应，即可求得系统在不同输入时产生的输出。

因此，求解离散系统的单位响应$h(k)$对我们进行离散系统的分析也同样具有非常重要的意义。

MATLAB 为用户提供了专门用于求离散系统单位响应，并绘制其时域波形的函数 impz()。在调用函数 impz()时，与连续系统一样，我们也需要用向量来对离散系统进行表




示。

设描述离散系统的差分方程为：

$$\sum_{i=0}^{N} a_i y(k-i) = \sum_{j=0}^{M} b_j f(k-j)$$

则我们可以用向量 a 和 b 表示该系统，即：

$$a = [a_0, a_1, \cdots\cdots, a_{N-1}, a_N]$$

$$b = [b_0, b_1, \cdots\cdots, b_{M-1}, b_M]$$

例如，某差分方程为：

$$y(k) - y(k-1) - 2y(k-2) = f(k)$$

则表示该离散系统的对应向量应为：

```
a=[1 -1 -2];
b=[1];
```

在用向量来表示差分方程描述的离散系统时，缺项要用 0 来补齐。例如，对差分方程

$$y(k) - 8y(k-2) = f(k) - f(k-1)$$

则表示该离散系统的对应向量应为：

```
a=[1 0 8];
b=[1 -1];
```

函数 impz( )能绘出向量 a 和 b 定义的离散系统在指定时间范围内单位响应的时域波形，并能求出系统单位响应在指定时间范围内的数值解。Impz()函数有如下几种调用格式：

- impz(b,a)

该调用格式以默认方式绘出向量 a 和 b 定义的离散系统的单位响应的离散时间波形。例如，若描述某离散系统的差分方程为：

$$y(k) - y(k-1) + 0.9y(k-3) = f(k)$$

运行如下 MATLAB 命令：

```
a=[1 -1 0.9];
b=[1];
impz(b,a)
```

则绘出该离散系统的单位响应的时域波形，如图 7-13 所示。

- impz(b,a,n)



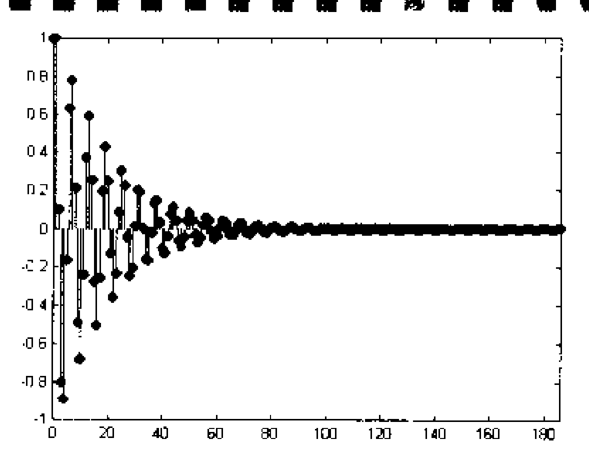


图 7-13　离散系统的单位响应一

该调用格式将绘出由向量 a 和 b 定义的离散系统在 0~n（n 必须为整数）离散时间范围内单位响应的时域波形。对上例，若运行如下命令：

```
impz(b,a,60)
```

则绘出系统在 0~60 取样点范围内单位响应的离散时间波形，如图 7-14 所示。

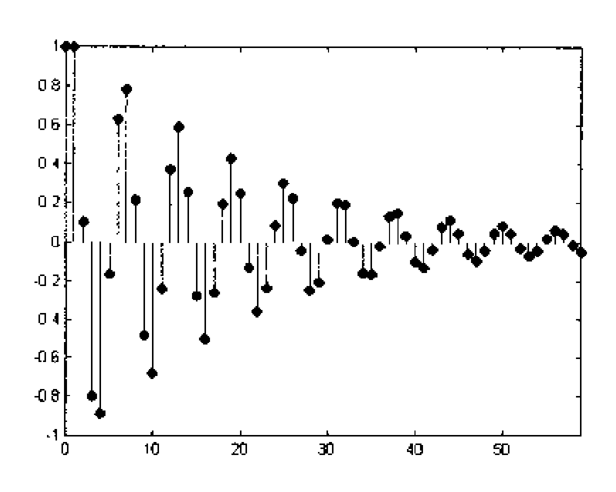


图 7-14　离散系统的单位响应二

- impz(b,a,n1:n2)

该调用格式将绘出由向量 a 和 b 定义的离散系统在 $n1$~$n2$（$n1$、$n2$ 必须为整数，且 $n1<n2$）离散时间范围内单位响应的时域波形。对上例，若运行如下命令：

```
impz(b,a,-10:40)
```

则绘出系统在-10~40 离散时间范围内单位响应的时域波形，如图 7-15 所示。

- y=impz(b,a,n1:n2)

该调用格式并不绘出系统单位响应的时域波形，而是求出向量 a 和 b 定义的离散系统在 $n1$~$n2$ 离散时间范围内的系统单位响应的数值解。对上例，若运行命令：

```
a=[1 -1 0.9];
b=[1];
y=impz(b,a,-5:10)
```



运行结果为：

```
y =
          0
          0
          0
          0
          0
     1.0000
     1.0000
     0.1000
    -0.8000
    -0.8900
    -0.1700
     0.6310
     0.7840
     0.2161
    -0.4895
    -0.6840
```

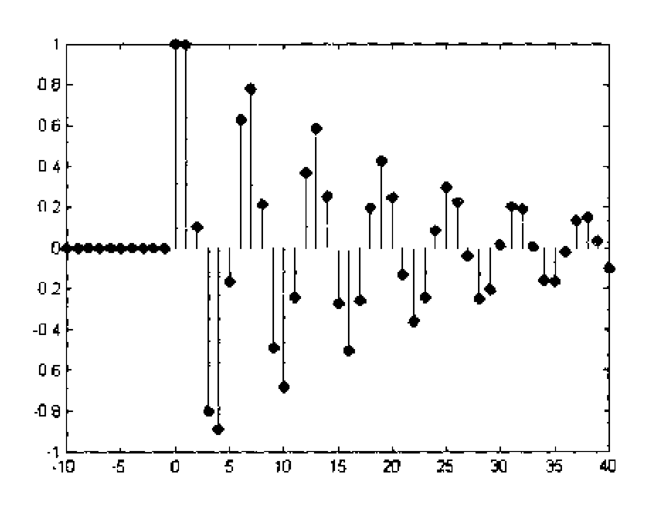


图 7-15　离散系统的单位响应

例 7-6：已知描述某离散系统的差分方程如下：

$$2y(k)-2y(k-1)+y(k-2)=f(k)+3f(k-1)+2f(k-2)$$

试用 MATLAB 绘出该系统 0~50 时间范围内单位响应的波形。

解：

首先用向量 a 和 b 表示出该离散系统，然后再调用 impz()函数即可解决此问题。实现上述过程的 MATLAB 命令如下：

```
a=[2 -2 1]
```

```
b=[1 3 2];
impz(b,a)
```

绘制的系统单位响应波形图如图 7-16 所示。

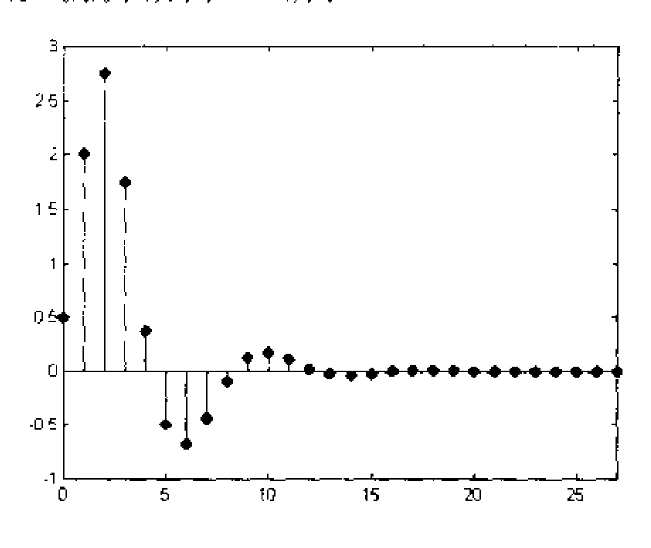


图 7-16 离散系统的单位响应四

7.5 利用 MATLAB 求 LTI 连续系统的响应

我们知道，LTI 连续系统可用如下所示的线性常系数微分方程来描述，

$$\sum_{i=0}^{N} a_i y^{(i)}(t) = \sum_{j=0}^{M} b_j f^{(j)}(t)$$

如果系统的输入信号及初始状态已知，我们便可用微分方程的经典时域求解方法，求出系统的响应。但对于高阶系统，手工计算这一问题的过程将会非常困难和繁琐。

MATLAB 的函数 lsim( )能对上述微分方程描述的 LTI 连续系统的响应进行仿真。lsim( )函数能绘制连续系统在指定的任意时间范围内系统响应的时域波形图，还能求出连续系统在指定的任意时间范围内系统响应的数值解。lsim()函数有如下两种调用格式：

- lsim(b,a,x,t)

在该调用格式中，a 和 b 是由描述系统的微分方程系数决定的表示该系统的两个行向量（与 7.3 节所述相同）。x 和 t 则是表示输入信号的行向量，其中 t 为表示输入信号时间范围的向量，x 则是输入信号在向量 t 定义的时间点上的取样值。例如命令：

```
t=0:0.01:10;
x=sin(t);
```

就定义了 0~10 秒范围内的正弦输入信号 sin(t)（取样时间间隔为 0.01 秒）。当取样时间间隔足够小时，向量 x 和 t 所定义的离散信号就是连续信号 sin(t)的较好的近似。

该调用格式将绘出由向量 b 和 a 所定义的连续系统在输入为向量 x 和 t 所定义的信号时，系统的零状态响应的时域仿真波形，且时间范围与输入信号相同。例如，描述某连续



系统的微分方程为：

$$y(t)'' + 2y(t)' + y(t) = f(t)' + 2f(t)$$

若要求当输入信号为 $f(t) = e^{-2t}\varepsilon(t)$ 时该系统的零状态响应 $y(t)$，我们可通过如下 MATLAB 命令来实现：

```
a=[1 2 1];
b=[1 2];
p=0.5;                    %定义取样时间间隔
t=0:p:5;                  %定义时间范围
x=exp(-2*t);              %定义输入信号
lsim(b,a,x,t);            %对系统输出信号进行仿真
```

上述命令绘制的系统零状态响的仿真波形，如图 7-17 所示。

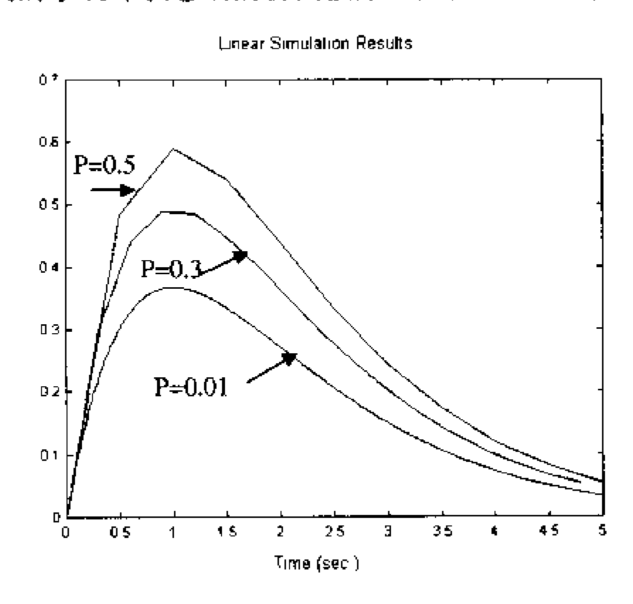


图 7-17　连续系统响应仿真

显然，函数 lsim( )对系统响应进行仿真的效果取决于向量 t 的时间间隔的密集程度。图 7-17 绘出了上述系统在不同取样时间间隔时函数 lsim 仿真的情况，可见取样时间间隔越小则仿真效果越好。

- y=lsim(b,a,x,t)

与前面介绍的函数 impulse 和 step 一样，该调用格式并不绘出系统的零状态响应曲线，而是求出与向量 t 定义的时间范围相一致的系统零状态响应的数值解。

7.6　利用 MATLAB 求 LTI 离散系统的响应

MATLAB 为用户提供了求 LTI 离散系统响应的专用函数 filter()。该函数能求出由差分方程描述的离散系统在指定时间范围内的输入序列时所产生的响应序列的数值解。

filter()函数的调用格式如下：

filter(b,a,x)

其中 b 和 a 是由描述系统的差分方程的系数决定的表示离散系统的两个行向量（与 7.4 节相同），x 是包含输入序列非零样值点的行向量。则上述命令将求出系统在与 x 的取样时间点相同的输出序列样值，即输出向量 y 包含了与输入向量 x 所在样本同一区间上的样本。

例 7-7：已知描述离散系统的差分方程为：

$$y(k)-0.25y(k-1)+0.5y(k-2)=f(k)+f(k-1)$$

且知该系统输入序列为 $f(k)=(\frac{1}{2})^k\varepsilon(k)$

试用 MATLAB 实现下列分析过程：

画出输入序列的时域波形；

求出系统零状态响应在 0~20 区间的样值；

画出系统的零状态响应波形图。

解：

我们可以调用 filter()函数来解决此问题。实现这一过程的 MATLAB 命令如下：

```
a=[1 -0.25 0.5];
b=[1 1];
t=0:20;
x=(1/2).^t;
y=filter(b,a,x)
subplot(2,1,1)
stem(t,x)
title('输入序列')
subplot(2,1,2)
stem(t,y)
title('响应序列')
```

程序运行结果为：

```
y =
  Columns 1 through 7
    1.0000    1.7500    0.6875   -0.3281   -0.2383    0.1982    0.2156
  Columns 8 through 14
   -0.0218   -0.1015   -0.0086    0.0515    0.0187   -0.0204   -0.0141
  Columns 15 through 21
    0.0069    0.0088   -0.0012   -0.0047   -0.0006    0.0022    0.0008
```

绘制的系统输入及响应序列波形图如图 7-18 所示。

需要注意的是，filter()函数将向量 x 以外的输入序列样值均视为零，这样若输入序列为时间无限长序列，则用 filter()函数计算系统响应时，在输出向量 y 的边界样点上，将会产生一定的偏差。

利用 filter()函数，我们还可以方便地求出由差分方程描述的离散系统的阶跃响应，此时，只需将输入信号定义为单位阶跃序列 $\varepsilon(k)$ 即可.

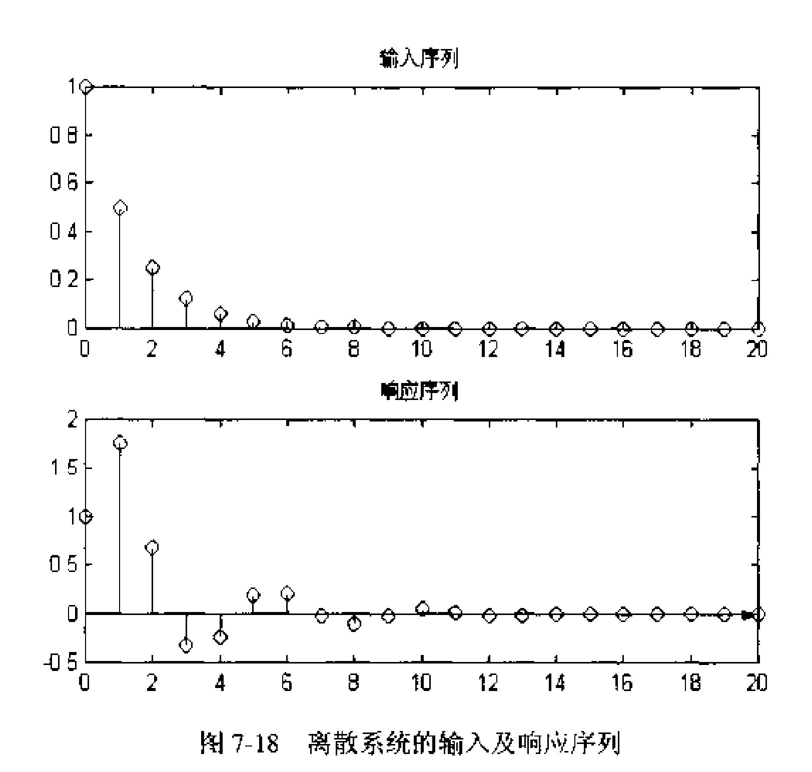


图 7-18 离散系统的输入及响应序列

例 7-8：已知离散系统的差分方程为：

$$y(k)+y(k-1)+\frac{1}{4}y(k-2)=f(k)$$

试用 MATLAB 绘出该系统单位阶跃响应 $g(k)$ 的时域波形.

解：

只要将输入序列定义为单位阶跃序列 $\varepsilon(k)$，再调用 filter()函数和 stem()函数即可解决此问题，实现这一过程的 MATLAB 命令如下：

```
a=[1 1 1/4];
b=[1];
t=0:15;
x=ones(1,length(t));
y=filter(b,a,x);
stem(t,y);
title('离散系统阶跃响应')
xlabel('k');
ylabel('g(k)')
```

绘制的系统阶跃响应序列波形如图 7-19 所示。

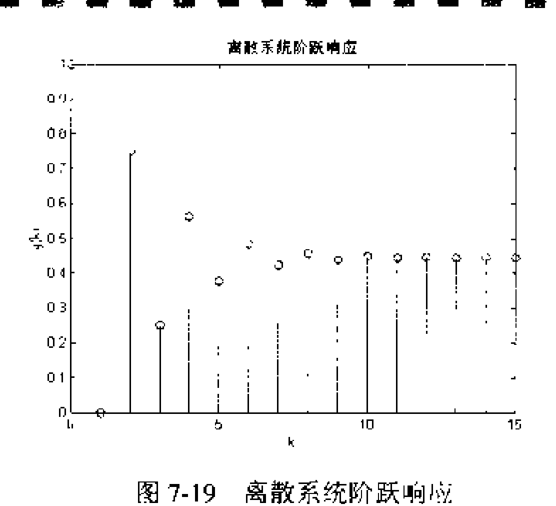
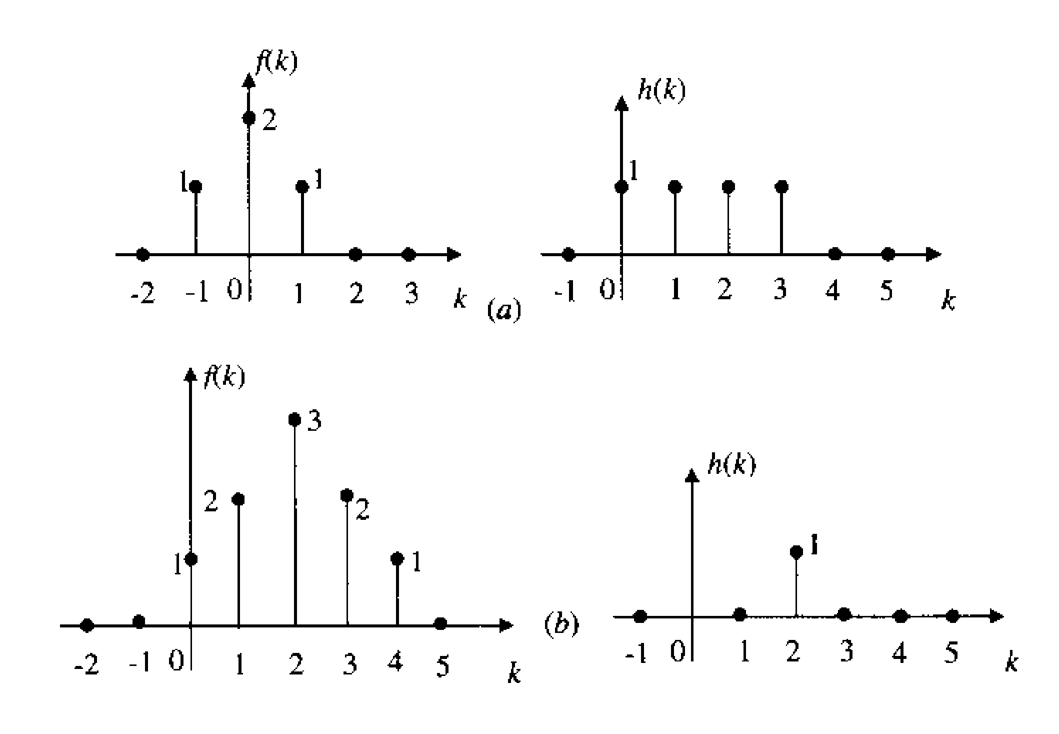


图 7-19 离散系统阶跃响应

上机练习题

1．已知某线性时不变离散系统的单位响应 $h(k)$ 和激励 $f(k)$ 分别如下图（a）和（b）所示，试用解析方法求系统的零状态响应 $y(k)$，绘出其时域波形，并用 MATLAB 实现上述过程，验证结果是否相同。



2．已知各离散序列的波形如下图所示，试用 MATLAB 求下列卷积和，并绘出卷积和序列的时域波形。

（1）$f_1(k)*f_2(k)$

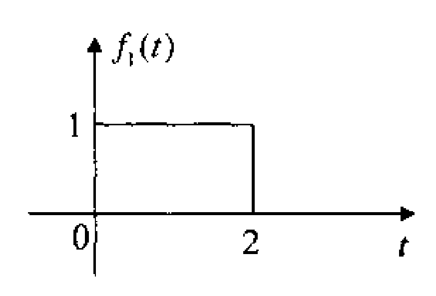


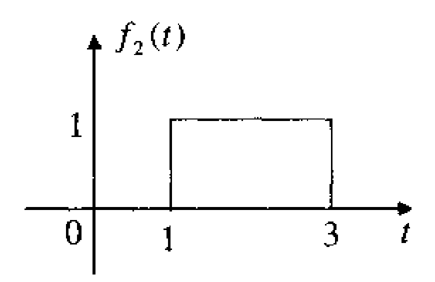
（2） $f_2(k) * f_3(k)$

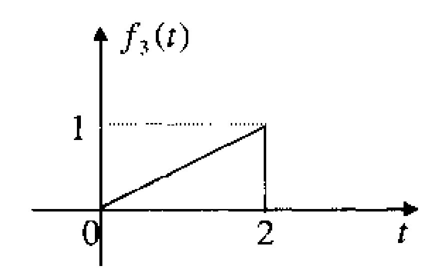
（3） $f_3(k) * f_4(k)$

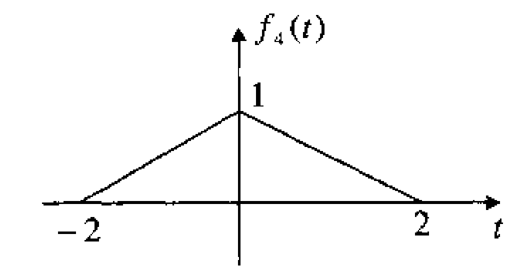
（4） $[f_2(k) - f_1(k)] * f_3(k)$

3. 已知各连续信号的波形如下图所示，试用解析方法求下列卷积，并用 MATLAB 绘出卷积信号波形，将其与解析计算结果进行比较。

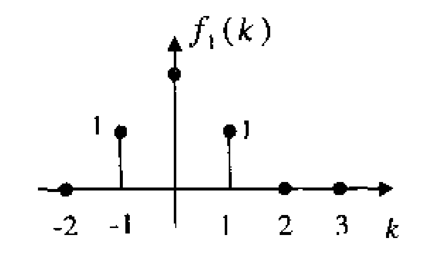


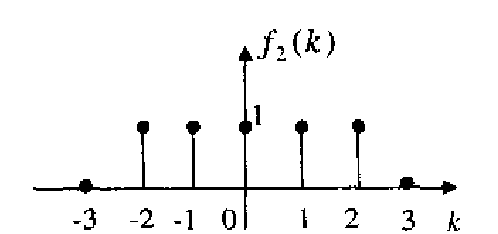


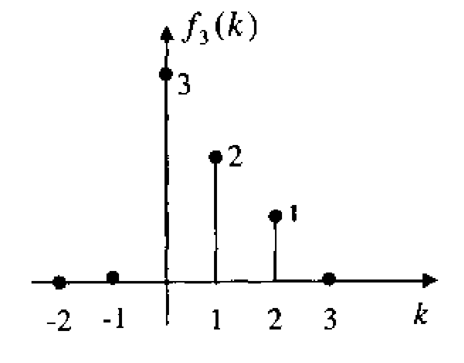


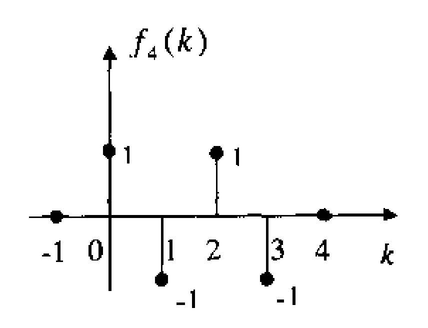



（1） $f_1(t) * f_2(t)$











（2） $f_1(t)*f_3(t)$

（3） $f_2(t)*f_3(t)$

（4） $f_1(t)*f_4(t)$

（5） $f_3(t)*f_4(t)$

4．用 MATLAB 绘出下列信号的卷积积分 $f_1(t)*f_2(t)$ 的时域波形。

（1） $f_1(t)=t\varepsilon(t)$、$f_2(t)=\varepsilon(t)$

（2） $f_1(t)=\varepsilon(t)-\varepsilon(t-4)$， $f_2(t)=\sin(\pi t)\varepsilon(t)$

（3） $f_1(t)=e^{-2t}\varepsilon(t)$， $f_2(t)=e^{-t}\varepsilon(t)$

（4） $f_2(t)=e^{-t}\varepsilon(t)$， $f_2(t)=\varepsilon(t)$

5．已知描述系统的微分方程如下，试用 MATLAB 求系统在 0~10 秒范围内冲激响应和阶跃响应的数值解，并绘出系统冲激响应和阶跃响应的时域波形。

（1） $y''(t)+3y'(t)+2y(t)=f(t)$

（2） $y''(t)+2y'(t)+2y(t)=f'(t)$

（3） $y''(t)+y'(t)+y(t)=f'(t)+f(t)$

（4） $y''(t)+8y(t)=f(t)$

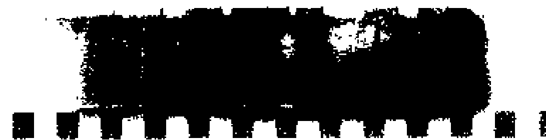
6．已知描述系统的微分方程和激励信号 $f(t)$ 如下，试用解析方法求系统的零状态响应 $y(t)$，并用 MATLAB 绘出系统零状态响应的时域仿真波形，验证结果是否相同。

（1） $y''(t)+4y'(t)+3y(t)=f(t)$， $f(t)=\varepsilon(t)$

（2） $y''(t)+4y'(t)+4y(t)=f'(t)+3f(t)$， $f(t)=e^{-t}\varepsilon(t)$

（3） $y''(t)+2y'(t)+2y(t)=f'(t)$， $f(t)=\varepsilon(t)$

7．已知描述离散系统的差分方程如下，试用 MATLAB 求出系统 0~20 时间样点的单



位序列响应和阶跃响应的数值解，并绘出其序列波形图。

（1） $y(k)+2y(k-1)=f(k-1)$

（2） $y(k)-y(k-2)=f(k)$

（3） $y(k)+y(k-1)+\frac{1}{4}y(k-2)=f(k)$

（4） $y(k)+4y(k-2)=f(k)$

8．试用 MATLAB 求下列差分方程描述的离散系统的零状态响应，并绘出零状态响应的时域波形。

（1） $y(k)-2y(k-1)=f(k)$，　$f(k)=\varepsilon(k)$

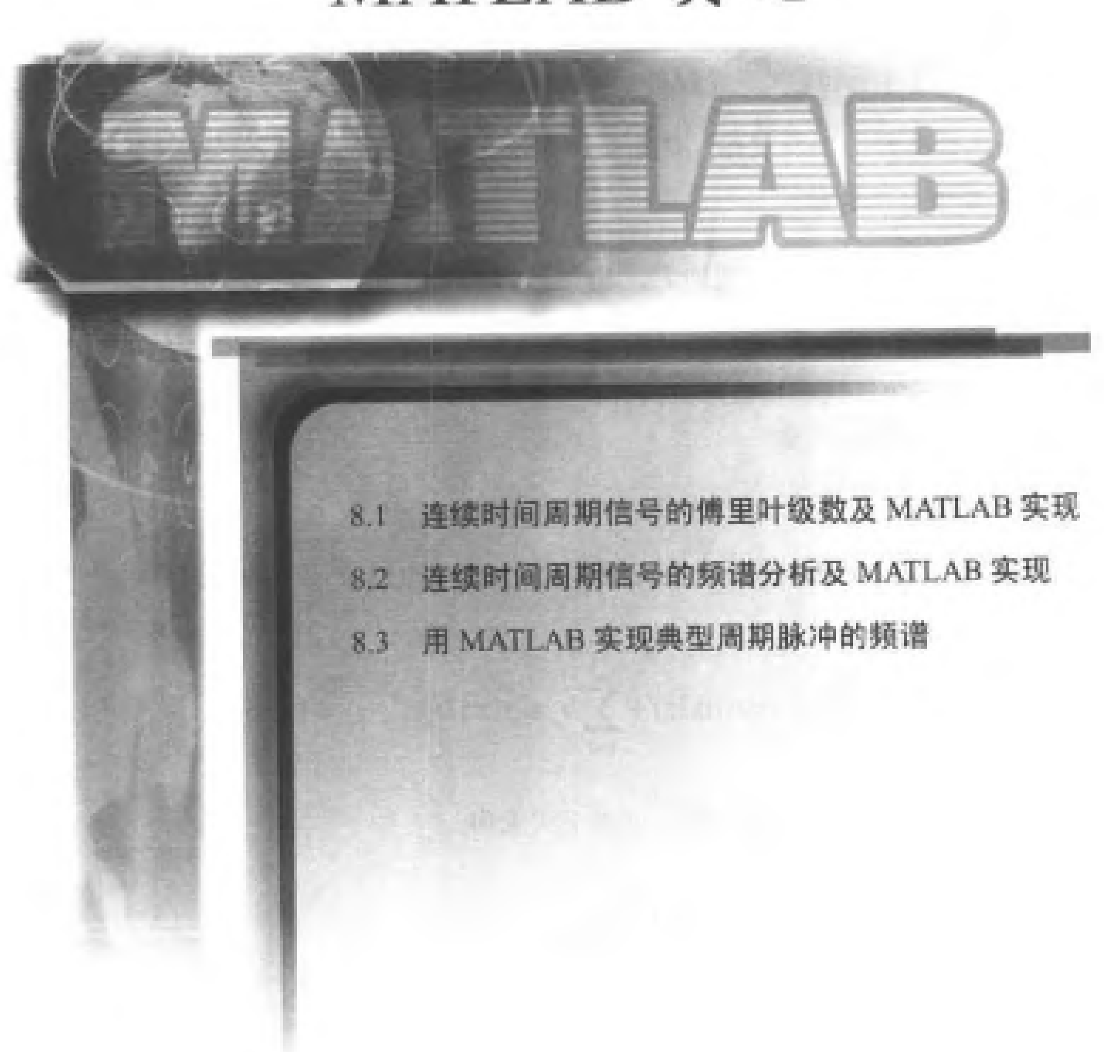
（2） $y(k)+2y(k-1)=f(k)$，　$f(k)=(\frac{1}{4})^k\varepsilon(k)$

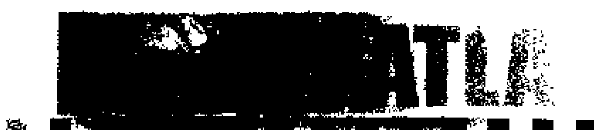
（3） $y(k)+2y(k-1)+y(k-2)=f(k)$，　$f(k)=(\frac{1}{4})^k\varepsilon(k)$



日期：________　姓名：________　备注：________

第 8 章　周期信号频域分析及 MATLAB 实现





8.1 连续时间周期信号的傅里叶级数及 MATLAB 实现

8.1.1 连续时间周期信号的傅里叶级数——CTFS

周期信号是定义在$(-\infty,+\infty)$区间，按一定时间间隔（周期 T）不断重复的信号。它可表示为：

$$f(t)=f(t+mT)$$

式中 m 为任意整数，T 为周期，周期的倒数称为该信号的频率。

1. 连续时间周期信号的分解

设有周期信号 $f(t)$，它的周期为 T，角频率 $\Omega=2\pi f=\dfrac{2\pi}{T}$，且满足狄里赫里条件，则该周期信号可以展开成傅里叶级数，即可表示为一系列不同频率的正弦或复指数信号之和。傅里叶级数有三角形式和指数形式两种。

1）三角形式的傅里叶级数

三角形式的傅里叶级数为：

$$f(t)=\frac{a_0}{2}+a_1\cos(\Omega t)+a_2\cos(2\Omega t)+\cdots+b_1\sin(\Omega t)+b_2\sin(2\Omega t)+\cdots$$

$$=\frac{a_0}{2}+\sum_{n=1}^{\infty}a_n\cos(n\Omega t)+\sum_{n=1}^{\infty}b_n\sin(n\Omega t)\qquad n=1,2,3,\cdots \tag{8-1}$$

式中系数 a_n、b_n 称为傅里叶系数，可由下式求得。

$$a_0=\frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)\mathrm{d}t \tag{8-2}$$

$$a_n=\frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)\cos(\Omega t)\mathrm{d}t \tag{8-3}$$

$$b_n=\frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)\sin(\Omega t)\mathrm{d}t \tag{8-4}$$

如果将（8-1）式中同频率的正弦和余弦分量合并，则三角形式的傅里叶级数可表示为：



$$f(t)=\frac{A_0}{2}+\sum_{n=1}^{\infty}A_n\cos(n\Omega t+\varphi_n)\qquad n=1,2,3,\cdots \tag{8-5}$$

上式中

$$\left.\begin{aligned}A_0&=a_0\\A_n&=\sqrt{a_n^2+b_n^2},\qquad n=1,2,\ldots\\\varphi_n&=-\arctan\frac{a_n}{b_n}\end{aligned}\right\}\tag{8-6}$$

$$\left.\begin{aligned}a_0&=A_0\\a_n&=A_n\cos\varphi_n,\qquad n=1,2,\ldots\\b_n&=-A_n\sin\varphi_n,\end{aligned}\right\}$$

由式（8-2）~（8-4）可见，傅里叶系数a_n和b_n都是n或（$n\Omega$）的函数，其中a_n是n（或$n\Omega$）的偶函数，即有$a_{-n}=a_n$；而b_n是n或（$n\Omega$）的奇函数，即有$b_{-n}=-b_n$。

由式（8-6）可见，A_n是n或（$n\Omega$）的偶函数，即有$A_{-n}=A_n$；而φ_n是n或（$n\Omega$）的奇函数，即有$\varphi_{-n}=-\varphi_n$。

式（8-5）表明，任何满足狄里赫里条件的周期信号可分解为一系列不同频率的余弦（或正弦）分量的叠加。其中第一项$A_0/2$是常数项，它是周期信号中所包含的直流分量；第二项$A_1\cos(\Omega t+\varphi_1)$称为基波或一次谐波，它的角频率与原周期信号相同，$A_1$是基波振幅，$\varphi_1$是基波初相角；第三项$A_2\cos(2\Omega t+\varphi_2)$称为二次谐波，它的频率是基波频率的二倍，$A_2$是二次谐波振幅，$\varphi_2$是其初相角；依此类推，还有三次、四次等谐波。一般而言，$A_n\cos(n\Omega t+\varphi_n)$称为$n$次谐波，$A_n$是$n$次谐波振幅，$\varphi_n$是其初相角。式（8-5）表明，周期信号可以分解为各次谐波分量的叠加。

2）指数形式的傅里叶级数

指数形式的傅里叶级数表达式为：

$$f(t)=\sum_{n=-\infty}^{\infty}F_n e^{jn\Omega t},\qquad n=0,\pm1,\pm2,\pm3,\cdots \tag{8-7}$$

即周期信号可分解为一系列不同频率的虚指数信号之和，式中F_n称为傅里叶复系数，可由下式求得：

$$F_n=\frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)e^{-jn\Omega t}\mathrm{d}t \tag{8-8}$$

3）三角形式和指数形式傅里叶级数及各系数间的关系

傅里叶级数的指数形式和三角形式是等价的，其系数可互相转换。表 8-1 综合了三角形式和指数形式傅里叶级数及其系数，以及各系数间的关系。

表 8-1　周期函数展开为傅里叶级数

形　式	指数形式	三角函数形式
展开式	$f(t)=\sum_{n=-\infty}^{\infty}F_n e^{jn\Omega t}$ $F_n=\lvert F_n\rvert e^{j\varphi_n}$	$f(t)=\frac{a_0}{2}+\sum_{n=1}^{\infty}a_n\cos(n\Omega t)$ $+\sum_{n=1}^{\infty}b_n\sin(n\Omega t)$ $=\frac{A_0}{2}+\sum_{n=1}^{\infty}A_n\cos(n\Omega t+\varphi_n)$
傅里叶系数	$F_n=\frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)e^{-jn\Omega t}\mathrm{d}t$ $n=0,\pm1,\pm2,\pm3,\cdots$	$a_n=\frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)\cos(\Omega t)\mathrm{d}t$ $n=0,1,2,3,\cdots$ $b_n=\frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)\sin(\Omega t)\mathrm{d}t$ $n=1,2,3,\cdots$ $A_n=\sqrt{a_n^2+b_n^2}$ $\varphi_n=-\arctan\frac{a_n}{b_n}$
系数间的关系	$F_n=\frac{1}{2}A_n e^{j\varphi_n}=\frac{1}{2}(a_n-jb_n)$ $\lvert F_n\rvert=\frac{1}{2}A_n=\frac{1}{2}\sqrt{a_n{}^2+b_n{}^2}$ 是 n 的偶函数 $\varphi_n=-\arctan\frac{b_n}{a_n}$ 是 n 的奇函数	$a_n=A_n\cos\varphi_n=F_n+F_{-n}$ 是 n 的偶函数 $b_n=-A_n\sin\varphi_n=j(F_n-F_{-n})$ 是 n 的奇函数 $A_n=2\lvert F_n\rvert$

2. 连续时间周期信号的傅里叶综合

任何满足狄里赫里条件的周期信号，可以表示成式（8-1）或（8-7）的和式形式，（8-1）或（8-7）式常称为 CTFS 综合公式。

一般来说，傅里叶级数系数有无限个非零值，即任何具有有限个间断点的周期信号都一定有一个无限项非零系数的傅里叶级数表示。但对数值计算来说，这是无法实现的。在实际的应用中，我们可以用有限项的傅里叶级数求和来逼近。即对有限项和：

$$f(t)=\sum_{n=-N}^{N}F_n e^{jn\Omega t}$$

$$=\frac{a_0}{2}+\sum_{n=1}^{N}a_n\cos(n\Omega t)+\sum_{n=1}^{N}b_n\sin(n\Omega t) \qquad (8\text{-}9)$$



当 N 值取得较大时，上式就是原周期信号 $f(t)$ 的一个很好的近似。（8-9）式常称做 $f(t)$ 的截断傅里叶级数表示。

MATLAB 的符号积分函数 int（）可以帮助我们求出连续时间周期信号的截断傅里叶级数及傅里叶表示。

求积指令 int 的具体使用格式如下：

- int f =int(f ,v)：给出符号表达式 f 对指定变量 v 的（不带积分常数的）不定积分。
- int f =int(f ,v ，a，b)：给出符号表达式 f 对指定变量 v 的定积分。

8.1.2　利用 MATLAB 实现周期信号的傅里叶级数分解与综合

MATLAB 强大的符号运算功能为我们进行周期信号的分析提供了强有力的工具。本节我们以周期矩形脉冲信号为例，来说明如何用 MATLAB 来实现周期信号的分解与综合过程。

例 8-1：周期矩形脉冲信号如图 8-1 所示，其幅度为 1，脉冲宽度为 $tao=1$，周期 $T=5*tao$，试用 MATLAB 求出该信号三角形式的傅里叶系数，并绘出各次谐波叠加的傅里叶综合波形图。

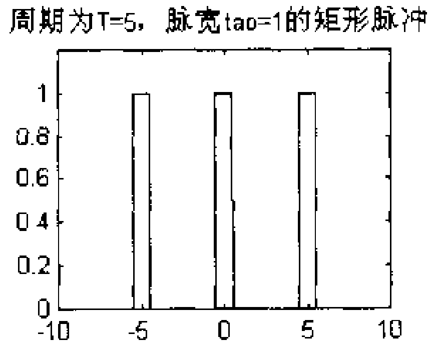


图 8-1　周期矩形脉冲函数

解：

本例采用三角形式傅里叶级数分解与综合形式，用式（8-2）~（8-4）求出傅里叶级数分解系数，运用 MATLAB 的符号运算功能，用式（8-9）实现信号的综合，谐波的阶数 $Nf=6$。

1．实现流程

利用 MATLAB 实现上述分析过程的流程如下：

（1）编写子函数 x=time_fun_x(t)，用符号表达式表示出周期信号在第一个周期内的符号表达式，并赋值给返回符号变量 x。

（2）编写子函数 y=time_fun_e(t)，求出该周期信号在绘图区间内的信号样值，并赋值给返回变量 y。

（3）编写求解信号傅里叶系数及绘制合成波形图的通用函数 CTFShchsym.m，该函数的流程如下：



a．调用函数 time_fun_x(t)，获取周期信号的符号表达式。

b．求出信号的傅里叶系数。

c．求出各次谐波。

d．绘制各次谐波叠加波形图。

e．调用函数 time_fun_e（t），绘制原信号波形图。

2．MATLAB 算法提示及说明

MATLAB 算法提示

（1）采用符号积分 int 求[0,T]内时间函数的三角级数展开系数：$a_0 = A_0$，$a_n = As$，$b_n = Bs$，即计算式（8-2）～（8-4）。

（2）用循环语句 *for...end* 求出三角级数展开系数 a_n，b_n 的数值，分别为 A_sym，B_sym。

（3）用 disp()语句输出三角级数展开系数 A_sym，B_sym。

（4）用傅里叶三角级数展开式（8-9）合成（综合）连续时间信号。

（5）化简表达式，据函数的奇偶性可知，若 $f(t)$ 为奇函数，则 $a_n = 0$；若 $f(t)$ 为偶函数，则 $b_n = 0$。以此化简三角级数展开式。

说明

本例的 CTFShchsym.m 函数文件有一定的通用性，用户只需编写好子函数 time_fun_x(t) 即可，但要注意，该函数是用符号表达式写成的。若要画出时间函数图形，用户需要另外编写一个子函数 y=time_fun_e(t)，因为在 MATLAB 中，只定义了单位阶跃信号 Heaviside 作为一个符号对象，而不能把 Heaviside 看做 MATLAB 的函数加以调用。

同理，在信号与系统中，另一个十分重要的函数——单位脉冲函数 Dirac(t)，它的使用方法可参照 Heaviside 进行。

最后给出的数值是由完全准确解取 32 位有效数字后的表示。

3．源程序

编写函数文件 CTFShchsym.m，这是一个计算连续时间周期信号的三角级数前 7 次展开系数，再用这 7 次系数合成原连续时间周期信号的程序，如下所示。

```
[CTFShchsym.m]
function[A_sym,B_sym]=CTFShchsym
%  采用符号计算法求一个周期内连续时间函数的三角级数展开系数，并绘出前七次
%  谐波合成波形。
%   函数的输出是数值量
%   a   被展开函数的时间区间的左端
%   Nf=6      谐波的阶数
%   Nn        输出数据的准确位数
%   A_sym     第 1 元素是直流项，其后元素依次是 1,2,3...次
%   谐波 cos 项展开系数
%   B_sym     第 2,3,4,...元素依次是 1,2,3...次谐波 sin 项展开系数
```



```
%    tao=1    tao/T=0.2
syms t   n    k   x
T=5;tao=0.2*T;a=0.5;
if nargin<4;Nf=6;end
if nargin<5;Nn=32;end
x=time_fun_x(t);               % 调用符号变量写成的周期矩形脉冲
A0=2*int(x,t,-a,T-a)/T;        % 求出三角函数展开系数 A0
As=int(2*x*cos(2*pi*n*t/T)/T,t,-a,T-a);  %求出三角函数展开系数 As
Bs=int(2*x*sin(2*pi*n*t/T)/T,t,-a,T-a);  %求出三角函数展开系数 Bs
A_sym(1)=double(vpa(A0,Nn));
                                          %  获取串数组 A0 所对应的 ASC2 码数值数组
for k=1:Nf
A_sym(k+1)=double(vpa(subs(As,n,k),Nn));
                                          %  获取串数组 A 所对应的 ASC II 码数值数组
B_sym(k+1)=double(vpa(subs(Bs,n,k),Nn));
                                         %  获取串数组 B 所对应的 ASC II 码数值数组
end
if nargout==0
c=A_sym;disp(c)     % 输出 c 为三角级数展开系数: 第 1 元素是直流项
                     % 其后元素依次是 1,2,3...次谐波 cos 项展开系数
d=B_sym;disp(d)     % 输出 d 为三角级数展开系数: 第 2,3,4,...元素
                     % 依次是 1,2,3...次谐波 sin 项展开系数
t=-8*a:0.01:T-a;
f1=0.2/2+0.1871.*cos(2*pi*1*t/5)+0.*sin(2*pi*1*t/5);  %  基波
f2=0.1514.*cos(2*pi*2*t/5)+0.*sin(2*pi*2*t/5);       % 2 次谐波
f3=0.1009.*cos(2*pi*3*t/5)+0.*sin(2*pi*3*t/5);       % 3 次谐波
f4=0.0468.*cos(2*pi*4*t/5)+0.*sin(2*pi*4*t/5);       % 4 次谐波
f5=-0.0312.*cos(2*pi*6*t/5)+0.*sin(2*pi*6*t/5);  % 6 次谐波
f6=f1+f2;            % 基波+2 次谐波
f7=f6+f3;            % 基波+2 次谐波+3 次谐波
f8=f7+f4+f5;         % 基波+2 次谐波+3 次谐波+4 次谐波+6 次谐波
subplot(2,2,1)
plot(t,f1),hold on
y=time_fun_e     % 调用符号函数写成的连续时间函数-周期矩形脉冲
plot(t,y)
title('周期矩形波的形成 －基波')
subplot(2,2,2)
plot(t,f6),hold on
y=time_fun_e
```

```
plot(t,y)
title('周期矩形波的形成—基波+2 次谐波')
subplot(2,2,3)
plot(t,f7),hold on
y=time_fun_e
plot(t,y)
title('周期矩形波的形成—基波+2 次谐波+3 次谐波')
subplot(2,2,4)
plot(t,f8),hold on
y=time_fun_e
plot(t,y)
title('周期矩形波的形成—基波+2 次谐波+3 次谐波+4 次谐波+6 次谐波')
end
%---
function x=time_fun_x(t)
% 该函数是 CTFShchsym.m 的子函数。它由符号变量和表达式写成
h=1;
x1=sym('Heaviside(t+0.5)')*h;
x=x1-sym('Heaviside(t-0.5)')*h;
%---
function y=time_fun_e
% 该函数是 CTFShchsym.m 的子函数，它形成周期矩形脉冲
a=0.5;T=5;h=1;tao=0.2*T;t=-8*a:0.01:T-a;
e1=1/2+1/2.*sign(t+tao/2);
e2=1/2+1/2.*sign(t-tao/2);
y=h.*(e1-e2);        %连续时间函数-周期矩形脉冲
```

4. 程序运行结果

调用 CTFShchsym..m 函数文件，即可绘出周期矩形波信号各次谐波的合成波形。

指令如下：在 MATLAB 命令窗口键入 CTFShchsym，并按回车键，即可绘出周期矩形波信号各次谐波的合成波形，如图 8-2 所示。

在 MATLAB 的指令窗口中，执行如下命令，就可得到三角级数展开系数为：

```
[A_sym,B_sym]= CTFShchsym (2)
A_sym =0.2000    0.3742    0.3027    0.2018    0.0935    0.0000    -0.0624
B_sym =   0        0         0         0         0         0          0
```

由图 8-2 可见，当它所包含的谐波分量越多时，合成波形越接近于原来的矩形波脉冲（图中虚线）。由图 8-2 还可以看到，合成波形所包含的谐波分量越多时，除间断点附近外，它越接近于原矩形波脉冲。在间断点附近，随着所含谐波次数的增加，合成波形的尖峰愈接近间断点，但尖峰幅度并未明显减少。可以证明，即使合成波形所含谐波次数



$n \to \infty$ 时，在间断点处仍有约 9%的偏差，这种现象称为吉布斯（Gibbs）现象。在傅里叶级数的项数取得很大时，间断点处尖峰下的面积非常小以致趋近于零，因而在均方的意义上合成波形同原波形的真值之间没有区别。

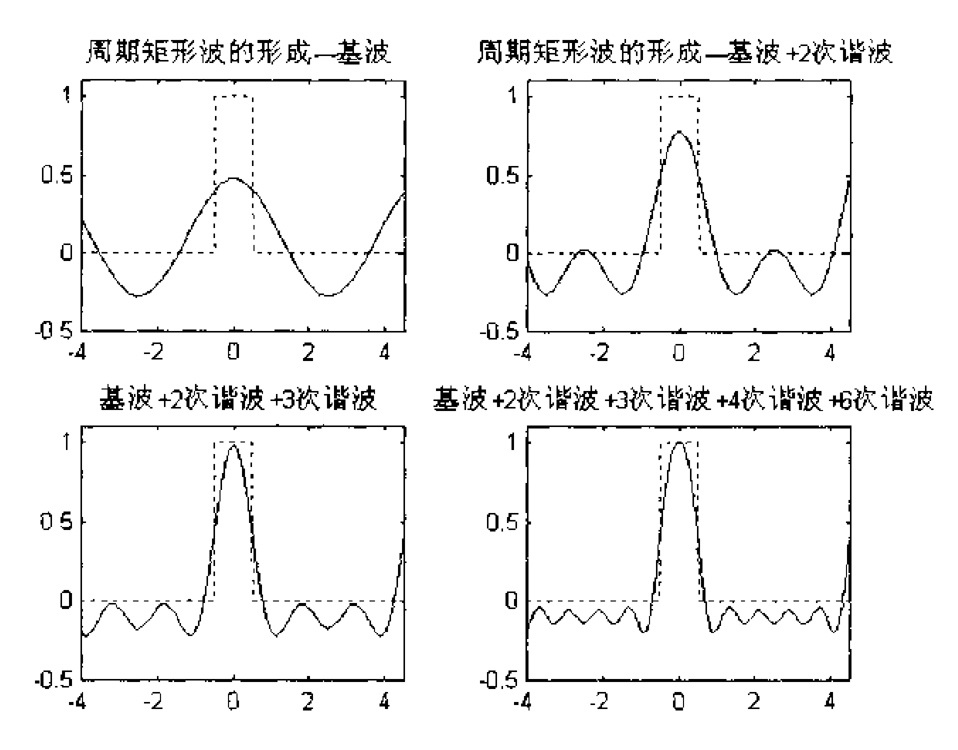


图 8-2　周期矩形脉冲函数的合成

（积分区间为[-a，T-a]，a 为被展开函数的时间区间的左端）

8.2　连续时间周期信号的频谱分析及 MATLAB 实现

8.2.1　连续时间周期信号的频谱分析

如前所述，周期信号可以分解成一系列正弦（余弦）信号或虚指数信号之和，即：

$$f(t) = \sum_{n=-\infty}^{\infty} F_n e^{jn\omega t}$$

$$= \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos(n\Omega t) + \sum_{n=1}^{\infty} b_n \sin(n\Omega t) \tag{8-10}$$

其中，$F_n = \frac{1}{2} A_n e^{j\varphi_n} = \frac{1}{2}(a_n - jb_n)$，　　　　(8-11)



$$\left.\begin{aligned} |F_n| &= \frac{1}{2}A_n = \frac{1}{2}\sqrt{a_n^{\ 2}+b_n^{\ 2}} \\ \varphi_n &= -\arctan\frac{b_n}{a_n} \end{aligned}\right\}\quad \begin{matrix}\text{幅度}\\ \text{相位}\end{matrix}$$

或

为了直观地表示出信号所含各分量的振幅 A_n 或 $|F_n|$ 随频率的变化情况，通常以角频率为横坐标，以各次谐波的振幅 A_n 或虚指数函数 $|F_n|$ 的幅度为纵坐标，画出如图 8-3 和图 8-4 所示的各谐波的振幅 A_n 或 $|F_n|$ 与角频率的关系图，称为周期信号的幅度（振幅）频谱，简称幅度谱。图中每条竖线代表该频率分量的幅度，称为谱线。各谱线顶点连线的曲线（如图中圆点所示）称为频谱包络线，它反映了各谐波分量幅度随频率变化的情况。图 8-3 中幅度谱为单边幅度谱（用 A_n 绘制的频谱）。图 8-4 中幅度谱为双边幅度谱（用 $|F_n|$ 绘制的频谱）。

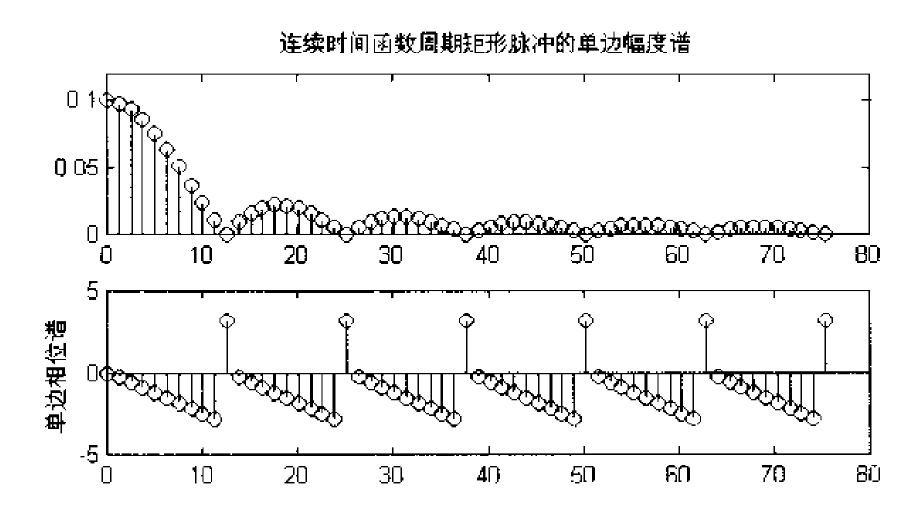


图 8-3　周期信号的幅度谱和相位谱（单边）

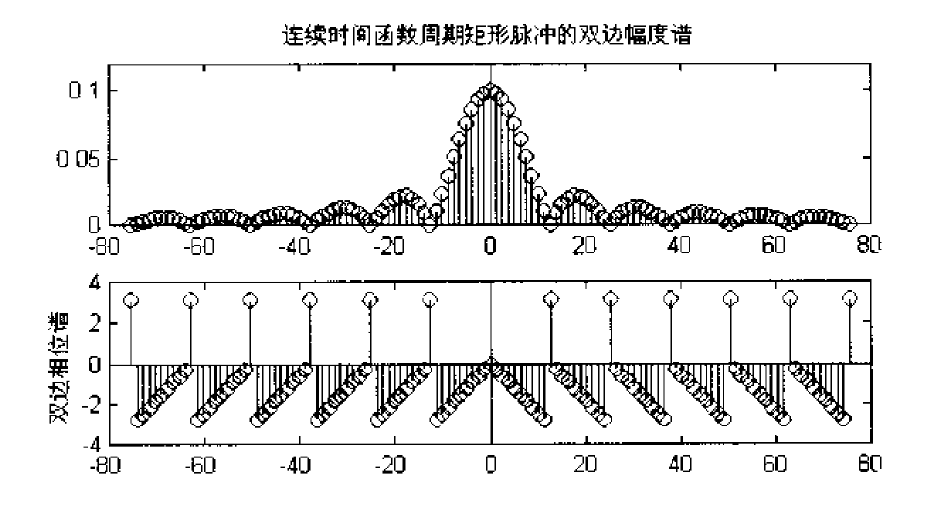


图 8-4　周期信号的幅度谱和相位谱（双边）

类似地，也可画出各谐波初相角 φ_n 与角频率的关系图，如图 8-3 和图 8-4 中各谐波



初相角 φ_n 与角频率的关系图，称为相位频谱，简称相位谱。图 8-3 中相位谱为单边相位谱，图 8-4 中相位谱为双边相位谱。如果 F_n 为实数，那么可用 F_n 的正负来表示 φ_n 为 0 或 π，也可把幅度谱和相位谱画在一张图上。

由图可见，周期信号的谱线只出现在频率为 0，Ω，2Ω 等原周期信号频率的整数倍的离散频率上，即周期信号的频谱是离散谱。

由此可见，周期信号频谱具有三个特点：

- 离散性，即谱线是离散的
- 谐波性，即谱线只出现在基波频率的整数倍上
- 收敛性，即谐波的幅度随谐波次数的增高而减小

MATLAB 的符号积分函数 int ()、一维数组的寻访概念和 fliplr () 可以帮助我们求出连续时间周期信号的频谱（傅里叶级数分析法）。

8.2.2 周期信号频谱分析及 MATLAB 实现

由上述分析可知，我们只要求出了周期信号傅里叶级数 A_n（或 F_n）及 φ_n，我们就可根据 A_n（或 F_n）及 φ_n 随频率 $n\Omega$（$\Omega=\dfrac{2\pi}{T}$）的变化关系画出信号的振幅和相位频谱。

本节仍以图 8-1 所示的周期矩形脉冲信号为例，来说明如何用 MATLAB 绘制周期信号的频谱图，并对周期信号的频谱特件进行分析。

例 8-2：试用 MATLAB 绘出如图 8-1 所示的周期矩形脉冲信号的振幅频谱。

解：

由于绘制频谱的前提是必须先求出周期信号的傅里叶系数，因此我们只需对例 8-1 给出的求周期信号傅里叶级数的函数 CTFShchsym..m 进行适当修改，即可编写出绘制周期信号频谱的通用函数。

需要注意的是，由于周期信号的频谱是离散的，故在绘制频谱时，我们采用的是 stem 命令而不是 plot 命令，下面介绍实现上述过程的通用程序 CTFStpshsym.m。

1. 实现流程

本例采用三角形式傅里叶级数分解形式，用式（8-2）~（8-4）求出傅里叶级数分解系数 a_n 和 b_n，再用式（8-11），即：

$$F_n=\frac{1}{2}A_n e^{j\varphi_n}=\frac{1}{2}(a_n-jb_n)$$

求出傅里叶复指数系数 F_n，并画出 F_n 的振幅频谱。谐波的阶数 Nf 可任意指定，本例指定 Nf =60。

利用 MATLAB 实现上述分析过程的流程如下：

（1）编写子函数 y=time_fun_s(t)，用符号表达式表示出周期信号在第一个周期内的符号表达式，并赋值给返回符号变量 y。

（2）编写子函数 x=time_fun_e，求出该周期信号在绘图区间内的信号样值，并赋值给返回变量 x。

（3）编写求解信号傅里叶复指数系数 F_n 及绘制频谱图的通用函数，该函数的流程如下：

a．调用函数 time_fun_s(t)，获取周期信号的符号表达式。

b．求出信号的三角级数形式的傅里叶级数展开系数 a_n 和 b_n。

c．求出信号的复指数形式的傅里叶级数展开系数 F_n。

d．绘制 F_n 的振幅频谱图。

e．调用函数 time_fun_e，绘制信号波形图。

2．MATLAB 算法提示

（1）采用符号积分 int 求[0,T]内时间信号的三角级数展开系数：$a_0 = A_0$，$a_n = As$，$b_n = Bs$。即计算式（8-2）~（8-4）。

（2）用循环语句 *for...end* 求出三角级数展开系数 a_n，b_n 的数值，并赋值给变量 *A_sym*（*k*+1），*B_sym*（*k*+1）。如图 8-5 所示。

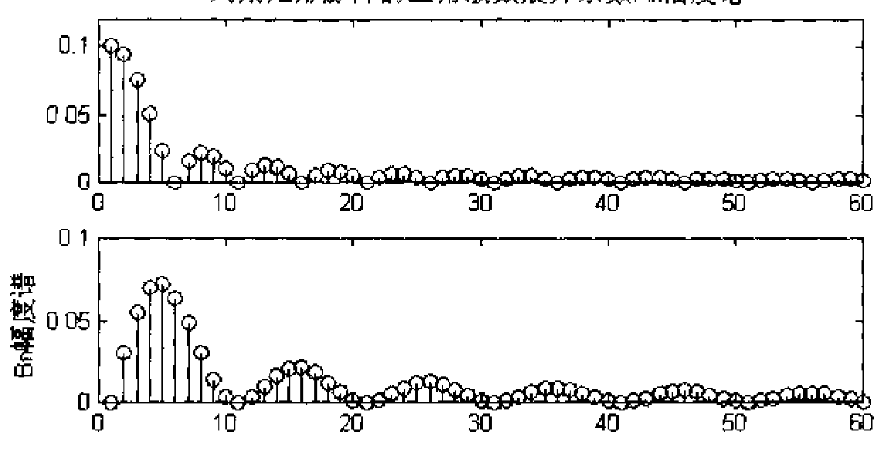


图 8-5　三角级数展开系数 a_n，b_n 频谱（*n*=*k*+1：*Nf*）

（3）从三角级数展开系数 a_n，b_n 得到复指数展开系数 F_n。

为了得到复指数展开系数 F_n，必须先求出三角形式的傅里叶级数展开系数 a_n 和 b_n，如图 8-5 所示，再根据（8-11）式求出 F_n。但要注意 a_n、b_n 和 F_n 的自变量取值情况，即：

$$F_n = \frac{1}{2}A_n e^{j\varphi_n} = \frac{1}{2}(a_n - jb_n)$$

$n = 0,\pm1,\pm2,\pm3,\cdots$	$n = 0,1,2,3,\cdots$

从上式的自变量取值情况及图 8-5 可见，三角级数展开系数 a_n，b_n 的变量 n 的取值范围为 n=0，1，2，3，…，N，而指数形式展开系数 F_n 的变量 n 的取值范围为 $n = 0,\pm1,\pm2,\cdots,\pm N$，为了从 a_n 和 b_n 得到 F_n，我们需要用到 MATLAB 的反折函数 fliplr

来实现频谱的反折。下面是运用 fliplr（）实现离散序列反折的示例。

例 8-3：已知一个原序列 a=[5 4 3 2 1]，采样位置：0，1，2，3，4。如图 8-6 所示，要组成原序列反折后再迭加原序列的一个新序列（采样位置：-4，-3，-2，-1，0，1，2，3，4）。程序如下：

```
n=0:4;            %原序列的位置
a=[5 4 3 2 1];
subplot(2,1,1), stem(n,a),
title('原序列')
b=fliplr(a);
k=-4:4;           %新序列的位置-4：4
c=[b,a(2:end)];
subplot(2,1,2), stem(k,c)
title('反折后的序列迭加原序列')
```

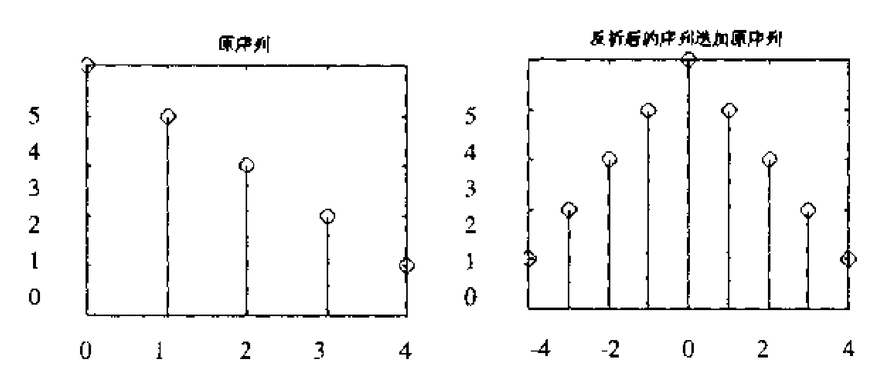
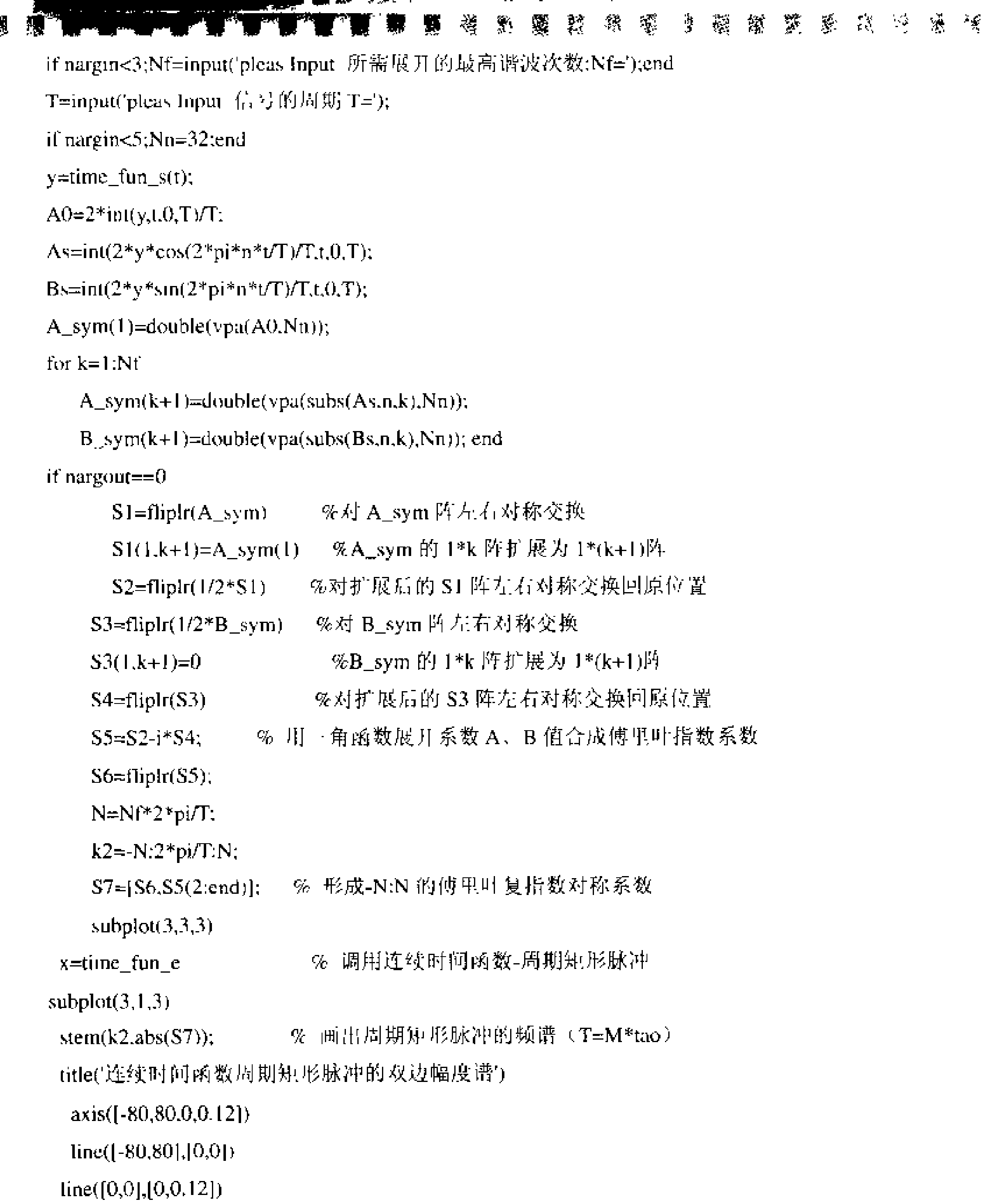


图 8-6　离散序列反折示例

3、源程序

编写函数文件 CTFStpshsym.m，如下所示：

```
{CTFStpshsym.m}
function [A_sym,B_sym]= CTFStpshsym
% 采用符号计算求[0,T]内时间函数的三角级数展开系数、并绘制其双边频谱
%             函数的输出为数值量
%   Nn        输出数据的准确位数
%   A_sym     第 1 元素是直流项，其后元素依次是 1,2,3...次谐波 cos 项展开系数
%   B_sym     第 2,3,4,...元素依次是 1,2,3...次谐波 sin 项展开系数
%   T        T=m*tao，信号周期
%   Nf        谐波的阶数
%   Nn        输出数据的准确位数
%  m (m=T/tao)周期与脉冲宽度之比，如 m=4,8,16,100 等
%  tao     脉宽:tao=T/m
syms t  n  y
```



```
if nargin<3;Nf=input('pleas Input 所需展开的最高谐波次数:Nf=');end
T=input('pleas Input 信号的周期 T=');
if nargin<5;Nn=32;end
y=time_fun_s(t);
A0=2*int(y,t,0,T)/T;
As=int(2*y*cos(2*pi*n*t/T)/T,t,0,T);
Bs=int(2*y*sin(2*pi*n*t/T)/T,t,0,T);
A_sym(1)=double(vpa(A0,Nn));
for k=1:Nf
    A_sym(k+1)=double(vpa(subs(As,n,k),Nn));
    B_sym(k+1)=double(vpa(subs(Bs,n,k),Nn)); end
if nargout==0
        S1=fliplr(A_sym)        %对 A_sym 阵左右对称交换
        S1(1,k+1)=A_sym(1)      %A_sym 的 1*k 阵扩展为 1*(k+1)阵
        S2=fliplr(1/2*S1)       %对扩展后的 S1 阵左右对称交换回原位置
      S3=fliplr(1/2*B_sym)      %对 B_sym 阵左右对称交换
      S3(1,k+1)=0               %B_sym 的 1*k 阵扩展为 1*(k+1)阵
      S4=fliplr(S3)             %对扩展后的 S3 阵左右对称交换回原位置
      S5=S2-i*S4;       % 用三角函数展开系数 A、B 值合成傅里叶指数系数
      S6=fliplr(S5);
      N=Nf*2*pi/T;
      k2=-N:2*pi/T:N;
      S7=[S6,S5(2:end)];    % 形成-N:N 的傅里叶复指数对称系数
      subplot(3,3,3)
  x=time_fun_e              % 调用连续时间函数-周期矩形脉冲
subplot(3,1,3)
  stem(k2,abs(S7));         % 画出周期矩形脉冲的频谱（T=M*tao）
  title('连续时间函数周期矩形脉冲的双边幅度谱')
   axis([-80,80,0,0.12])
   line([-80,80],[0,0])
  line([0,0],[0,0.12])
end
%---
function y=time_fun_s(t)
% 该函数是 CTFStpshsym.m 的子函数，它由符号变量和表达式写成
syms a a1
T=input('pleas Input 信号的周期 T=');
M=input('周期与脉冲宽度之比 M=');
A=1;tao=T/M;a=tao/2;
```



```
y1=sym('Heaviside(t+a1)')*A;
y=y1-sym('Heaviside(t-a1)')*A;
y=subs(y,a1,a);
y=simple(y);
%---
function x=time_fun_e
% 该函数是 CTFStpshsym.m 的子函数，它形成周期矩形脉冲。
%    t         是时间数组
%    T         是周期    duty=tao/T=0.2
T=5;t=-2*T:0.01:2*T;tao=T/5;
x=rectpuls(t,1);      %产生一个宽度 tao=1 的矩形脉冲
subplot(2,2,2)
plot(t,x)
hold on
x=rectpuls(t-5,1);    %产生一个宽度 tao=1 的矩形脉冲,中心位置在 t=5 处
plot(t,x)
hold on
x=rectpuls(t+5,1);  %产生一个宽度 tao=1 的矩形脉冲,中心位置在 t=-5 处
plot(t,x)
title('周期为 T=5，脉宽 tao=1 的矩形脉冲')
axis([-10,10,0,1.2])
```

4．程序运行结果及分析

1）程序运行结果

调用 CTFStpshsym.m 函数文件，即可绘出周期矩形波信号的频谱。指令如下：在 MATLAB 命令窗口键入 CTFStpshsym.m，并按回车键，命令窗口将出现：pleas Input 所需展开的最高谐波次数 Nf=？；本例输入 Nf=60，然后命令窗口将出现：pleas Input 信号的周期 T=？；本例输入周期 T=5，然后命令窗口又出现：周期与脉冲宽度之比 M=？；本例输入 M=5，即可绘出周期矩形波信号的频谱，如图 8-7 所示。

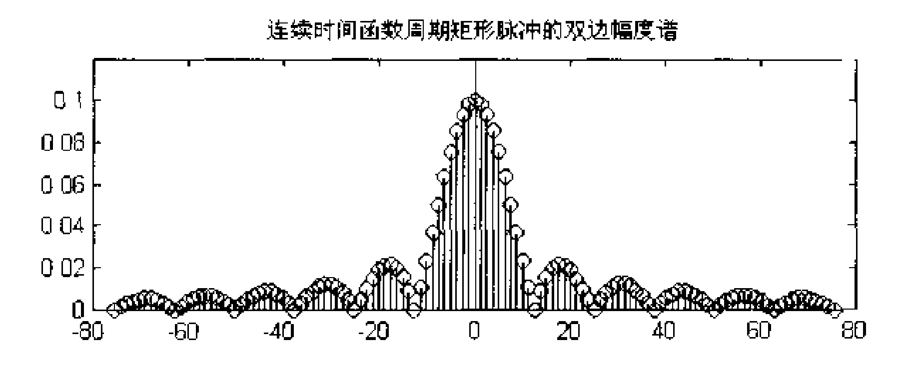
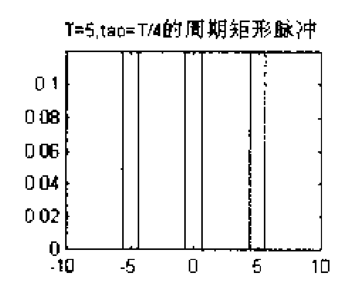


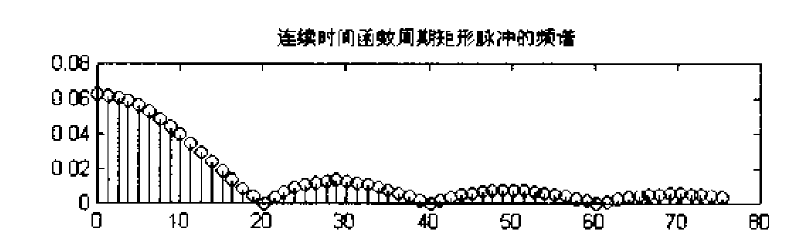
图 8-7　周期 T=5，脉宽 tao=1 的矩形脉冲及频谱图

由图 8-7 可见，周期矩形脉冲的频谱具有一般周期信号频谱的共同特点，即它们的频谱都是离散的。仅含有 $\omega = n\Omega$ 的各分量，其相邻两谱线的间隔是 $\Omega = 2\pi / T$ 。

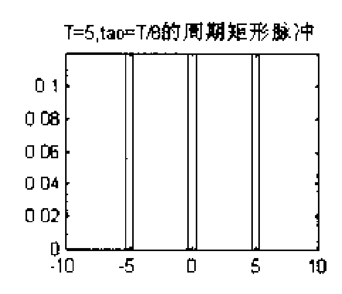
2）脉冲宽度与频谱的关系

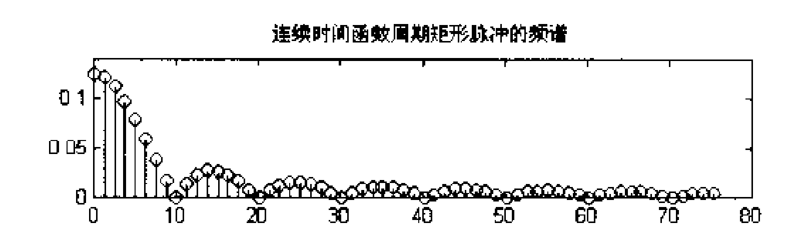
脉冲宽度与频谱的关系如图 8-8 所示。



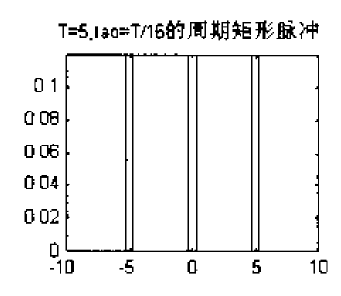


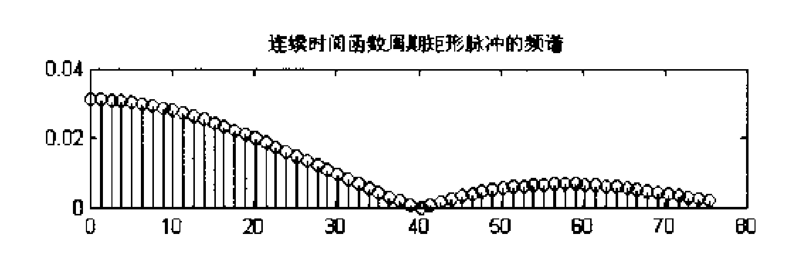
（*a*） 周期 T=5，脉冲宽度 tao=T/4=1.25，a=tao/2=0.625 的矩形脉冲频谱





（*b*） 周期 T=5，脉冲宽度 tao=T/8=0.625，a=tao/2=0.3125 的矩形脉冲频谱





（*c*） 周期 T=5，脉冲宽度 tao=T/16=0.3125，a=tao/2=0.15625 的矩形脉冲频谱

图 8-8 脉冲宽度与频谱的关系

修改上述绘制周期矩形脉冲双边频谱的函数文件[CTFStpshsym.m]，将其改为绘制周期矩形脉冲单边频谱的函数文件，并重复调用该函数文件，当窗口出现输入信号谐波次数 *Nf*，周期 *T*，周期与脉冲宽度之比 *M* 时，分别将信号设置为如表 8-2 所示的三种情况，则可绘出如表 8-2 所示各情况的信号波形及频谱图，如图 8-8 所示。

表 8-2 周期矩形脉冲脉宽取值参数表

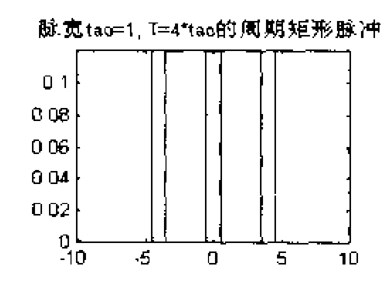
周期 *T*	脉冲宽度 *tao*	被展开函数的时间区间的左端 *a*=*tao*/2
5	tao=T/4=1.25	0.625
5	tao=T/8=0.625	0.3125
5	tao=T/16=0.3125	0.15625

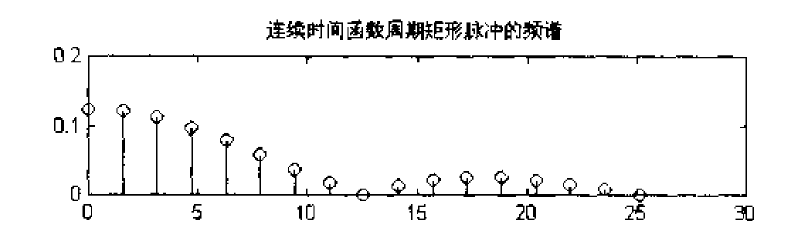


由图 8-8 可见，由于周期 T 相同，因而相邻谱线的间隔相同；脉冲宽度 *tao* 愈窄，其频谱包络线第一个零点的频率愈高，即信号带宽愈宽，频带内所含的分量愈多。可见，信号的频带宽度与脉冲宽度 *tao* 成反比。

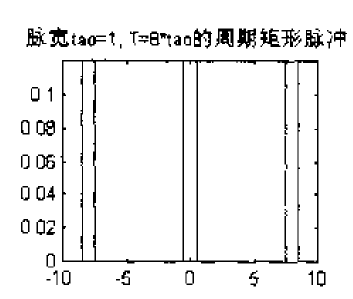
3）周期与频谱的关系

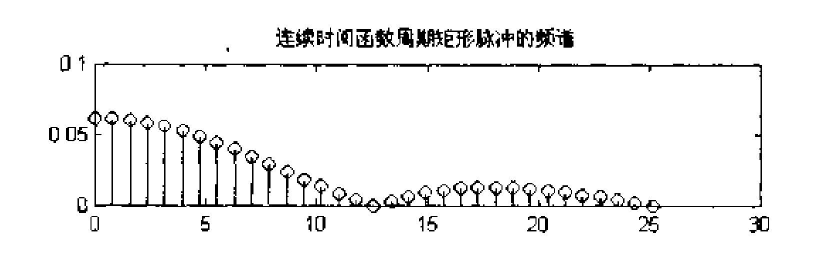
周期与频谱的关系如图 8-9 所示。



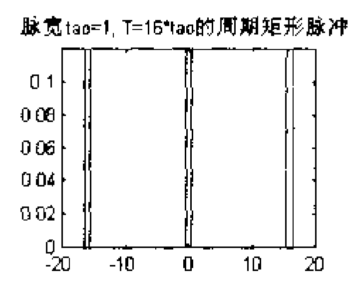


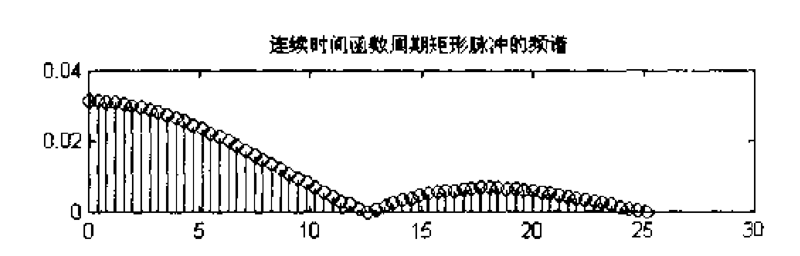
（*a*）　脉冲宽度 *tao*=1，周期 *T*=4**tao* =4 的矩形脉冲频谱



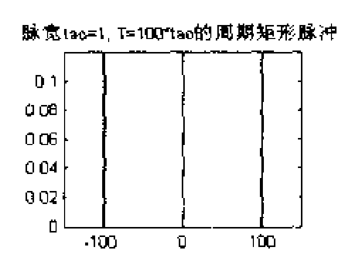


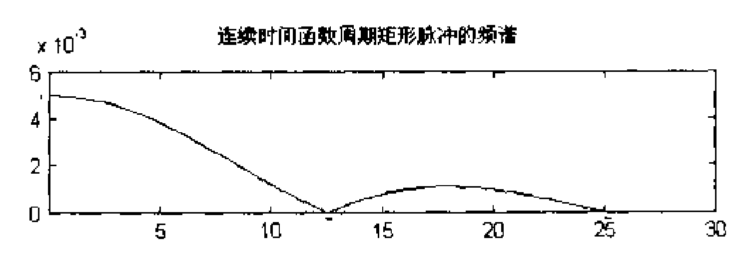
（*b*）　脉冲宽度 *tao*=1，周期 *T*=8**tao* =8 的矩形脉冲频谱





（*c*）　脉冲宽度 *tao*=1，周期 *T*=16**tao* =16 的矩形脉冲频谱





（*d*）　脉冲宽度 *tao*=1，周期 *T*=100**tao* =100 的矩形脉冲频谱

图 8-9　周期与频谱的关系

重复调用上述绘制周期矩形脉冲单边频谱的函数文件，当窗口出现输入信号谐波次数 *Nf*，周期 T，周期与脉冲宽度之比 M 时，分别将信号设置为表 8-3 所示的各种情况（即信号时域宽度保持不变），则可绘出表 8-3 所示各种情况下的信号波形及频谱图（如图 8-9





所示）

表 8-3 周期矩形脉冲周期取值参数表

脉冲宽度 *tao*	周期 *T*	被展开函数的时间区间的左端 a=tao/2
1	*T*=4**tao*=4	0.5
1	*T*=8**tao*=8	0.5
1	*T*=16**tao*=16	0.5
1	*T*=100**tao*=100	0.5

由图 8-9 可见，由于周期脉冲信号的时域宽度不变，这时频谱包络线的零点所在位置不变，而当周期增长时，相邻谱线的间隔减少，频谱变密。如果周期无限增长（这时就成为非周期信号），那么，相邻谱线的间隔将趋近于零，周期信号的离散频谱就过渡到非周期信号的连续频谱。随着周期的增长，各谐波分量的幅度也相应减少。脉冲周期 *T* 愈长，谱线间隔愈小，频谱越稠密；反之，则越稀疏。

8.3 用 MATLAB 实现典型周期脉冲的频谱

8.3.1 周期方波脉冲频谱的 MATLAB 实现

例 8-4：已知周期方波脉冲信号如图 8-10 所示，其幅度为 1，脉冲宽度‘占空比’：duty=1/2，周期 *T*=5 试用 MATLAB 编程绘出该周期信号的频谱。

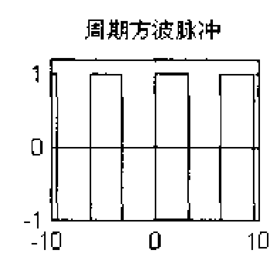


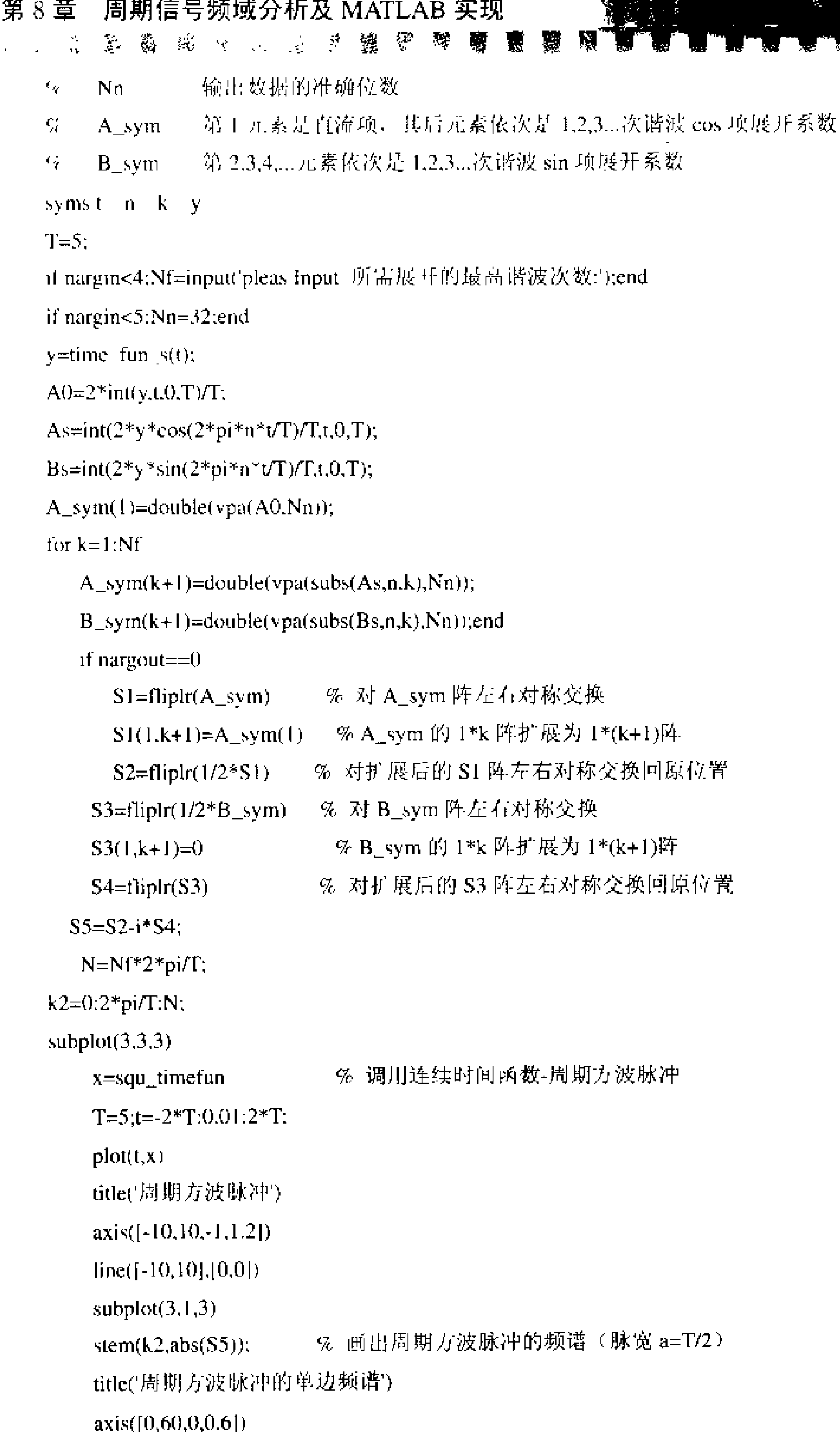
图 8-10 周期方波脉冲

解：

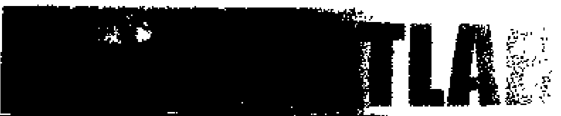
1．编写函数文件 CTFSdbfb.m

编写函数文件 CTFSdbfb.m 如下所示：

```
[CTFSdbfb.m]    %这是计算周期方波脉冲单边频谱的程序
function [A_sym,B_sym]=CTFSdbfb
% 采用符号计算求[0,T]内时间函数的三角级数展开系数。
%   T         输入信号的周期
%             函数的输入输出都是数值量
%   Nf        谐波的阶数
```



```
%    Nn        输出数据的准确位数
%    A_sym     第 1 元素是直流项，其后元素依次是 1,2,3...次谐波 cos 项展开系数
%    B_sym     第 2,3,4,...元素依次是 1,2,3...次谐波 sin 项展开系数
syms t  n  k  y
T=5;
if nargin<4;Nf=input('pleas Input 所需展开的最高谐波次数:');end
if nargin<5;Nn=32;end
y=time_fun_s(t);
A0=2*int(y,t,0,T)/T;
As=int(2*y*cos(2*pi*n*t/T)/T,t,0,T);
Bs=int(2*y*sin(2*pi*n*t/T)/T,t,0,T);
A_sym(1)=double(vpa(A0,Nn));
for k=1:Nf
   A_sym(k+1)=double(vpa(subs(As,n,k),Nn));
   B_sym(k+1)=double(vpa(subs(Bs,n,k),Nn));end
   if nargout==0
      S1=fliplr(A_sym)       % 对 A_sym 阵左右对称交换
      S1(1,k+1)=A_sym(1)     % A_sym 的 1*k 阵扩展为 1*(k+1)阵
      S2=fliplr(1/2*S1)      % 对扩展后的 S1 阵左右对称交换回原位置
     S3=fliplr(1/2*B_sym)    % 对 B_sym 阵左右对称交换
     S3(1,k+1)=0             % B_sym 的 1*k 阵扩展为 1*(k+1)阵
     S4=fliplr(S3)           % 对扩展后的 S3 阵左右对称交换回原位置
   S5=S2-i*S4;
   N=Nf*2*pi/T;
k2=0:2*pi/T:N;
subplot(3,3,3)
     x=squ_timefun           % 调用连续时间函数-周期方波脉冲
     T=5;t=-2*T:0.01:2*T:
     plot(t,x)
     title('周期方波脉冲')
     axis([-10,10,-1,1.2])
     line([-10,10],[0,0])
     subplot(3,1,3)
     stem(k2,abs(S5));       % 画出周期方波脉冲的频谱（脉宽 a=T/2）
     title('周期方波脉冲的单边频谱')
     axis([0,60,0,0.6])
     end
 %---
function y=time_fun_s(t)
```




```
% 该函数是 CTFSdbfb.m 的子函数。它由符号变量和表达式写成
syms a a1
T=5;a=T/2;
y1=sym('Heaviside(t)')*2-sym('Heaviside(t-a1)');
y=y1-sym('Heaviside(t+a1)');
y=subs(y,a1,a);
y=simple(y);
%---
function x=squ_timefun
% 该函数是 CTFSdbfb.m 的子函数,它由方波脉冲函数写成
%    t          时间数组
%    T          周期
%    duty: '占空比', 即信号为正的区域在一个周期内所占的百分比
T=5;t=-2*T:0.01:2*T;duty=50;
x=square(t,duty);
```

2. **程序运行结果**

调用函数 CTFSdbfb.m，即可绘出方波脉冲的频谱，如图 8-11 所示。

指令如下：在 MATLAB 命令窗口键入 CTFSdbfb，并按回车键，命令窗口将出现：pleas Input 所需展开的最高谐波次数 *Nf*=？，本例输入 *Nf*=60，即可绘出周期方波信号的单边频谱（如图 8-11 所示）。

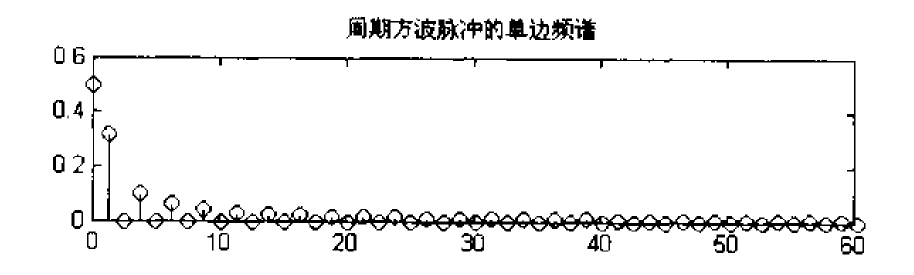


图 8-11 周期方波脉冲及单边频谱

将上述周期方波脉冲单边频谱的程序稍做改动，可绘出双边频谱，如图 8-12 所示。

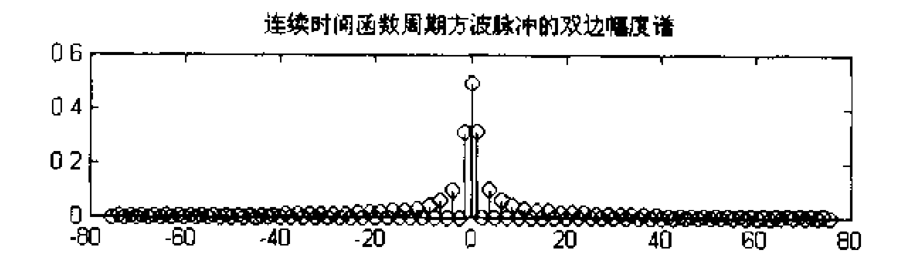


图 8-12 周期方波脉冲双边频谱



由图 8-12 可以看出，周期方波信号频谱与周期矩形脉冲信号具有相同的规律，由于方波的周期与脉宽比 $\frac{T}{\tau}=2$，因此频谱的第一个过零点内只有两根谱线。

8.3.2　周期三角波脉冲频谱的 MATLAB 实现

例 8-5：已知周期三角波脉冲如图 8-13 所示，周期 $T=5$，其幅度为±1，试用 MATLAB 绘出该信号的振幅频谱。

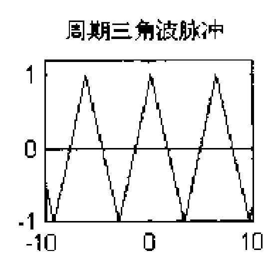


图 8-13　周期 T=5 的标准三角波

解：

1．产生三角波的函数

MATLAB 内置有产生三角波的函数 sawtooth(t)，其调用格式为：x= sawtooth(t,width)。根据 width 值的不同产生不同形状的三角波，参数 width 是 0～1 之间的标量，指定在一个周期之间最大值的位置，width 是该位置的横坐标和周期的比值。因而，当 width=0.5 时产生标准的对称三角波，当 width=1 时（可默认）产生锯齿波。

2．编写函数文件 CTFSsjbshbd.m

编写函数文件 CTFSsjbshbd.m 如下所示：

```
[CTFSsjbshbd.m]     %这是计算周期三角波脉冲双边频谱的程序
function [A_sym,B_sym]=CTFSsjbshbd
%  采用符号计算求[0,T]内时间函数的三角级数展开系数
%              函数的输入输出都是数值量
%  T           信号的周期
%  Nf          谐波的阶数
%  Nn          输出数据的准确位数
%  A_sym       第 1 元素是直流项，其后元素依次是 1,2,3...次谐波 cos 项展开系数
%  B_sym       第 2,3,4,...元素依次是 1,2,3...次谐波 sin 项展开系数
syms t  n  k  y
T=5;
if nargin<4;Nf=input('pleas Input 所需展开的最高谐波次数:');end
if nargin<5;Nn=32;end
y=time_fun_s(t);
```

```
A0=2*int(y,t,0,T)/T;
As=int(2*y*cos(2*pi*n*t/T)/T,t,0,T);
Bs=int(2*y*sin(2*pi*n*t/T)/T,t,0,T);
A_sym(1)=double(vpa(A0,Nn));
for k=1:Nf
    A_sym(k+1)=double(vpa(subs(As,n,k),Nn));
    B_sym(k+1)=double(vpa(subs(Bs,n,k),Nn));end
    if nargout==0
        S1=fliplr(A_sym)        % 对 A_sym 阵左右对称交换
        S1(1,k+1)=A_sym(1)      % A_sym 的 1*k 阵扩展为 1*(k+1)阵
        S2=fliplr(1/2*S1)       % 对扩展后的 S1 阵左右对称交换回原位置
      S3=fliplr(1/2*B_sym)      % 对 B_sym 阵左右对称交换
      S3(1,k+1)=0               % B_sym 的 1*k 阵扩展为 1*(k+1)阵
      S4=fliplr(S3)             % 对扩展后的 S3 阵左右对称交换回原位置
      S5=S2-i*S4;               % 用三角函数展开系数 A、B 值合成傅里叶指数系数
      S6=fliplr(S5);            % 对傅里叶指数复系数 S6 阵左右对称交换位置
      N=Nf*2*pi/T;
      k2=-N:2*pi/T:N;           % 形成-N:N 的变量
      S7=[S6,S5(2:end)];        % 形成-N:N 的傅里叶指数对称复系数
       subplot(3,3,3)
      x=sjb_timefun        % 调用连续时间函数-周期三角波脉冲
      T=5;t=-2*T:0.01:2*T;
      plot(t,x)
      title('连续时间函数-周期三角波脉冲')
      axis([-10,10,-1,1.2])
      line([-10,10],[0,0])
      subplot(3,1,3),
      stem(k2,abs(S7));  %画出周期三角脉冲的频谱（脉宽 a=T/2）
    title('连续时间函数周期三角脉冲的双边幅度谱')
    axis([-80,80,0,0.25])
    end
    %---
function y=time_fun_s(t)
% 该函数是 CTFSsjbshbd.m 的子函数。它由符号变量和表达式写成
syms a a1
T=5;a=T/2;
y1=sym('Heaviside(t+a1)')*(2*t/a1+1)+sym('Heaviside(t-a1)')*(2*t/a1-1);
y=y1-sym('Heaviside(t)')*(4*t/a1);
y=subs(y,a1,a);
```



```
y=simple(y);

%---
function x=sjb_timefun
% 该函数是 CTFSsjbshbd.m 的子函数。它由三角波脉冲函数写成。
T=5;t=-2*T:0.01:2*T;
x=sawtooth(t-2*T/3,0.5);
```

3．程序运行结果

调用函数 CTFSsjbshbd.m，即可绘出三角波的频谱，如图 8-14 所示。

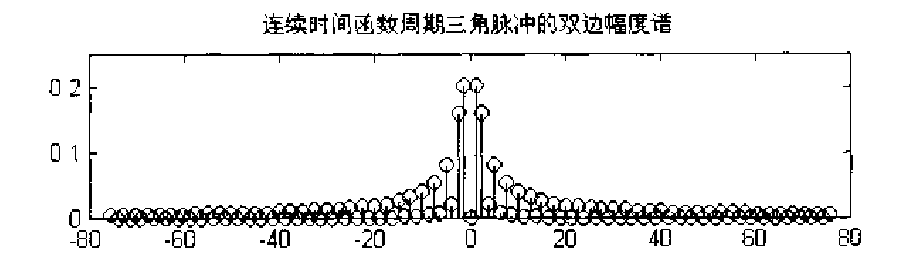


图 8-14　周期三角波双边脉冲频谱

指令如下：在 MATLAB 命令窗口键入 CTFSsjbshbd，并按回车键，命令窗口将出现 pleas Input 所需展开的最高谐波次数 *Nf*=？，本例输入 *Nf*=60，即可绘出周期三角波信号的频谱图。程序运行结果如图 8-14 所示。

将上述周期三角波脉冲双边频谱的程序稍做改动，可求出单边频谱，如图 8-15 所示。

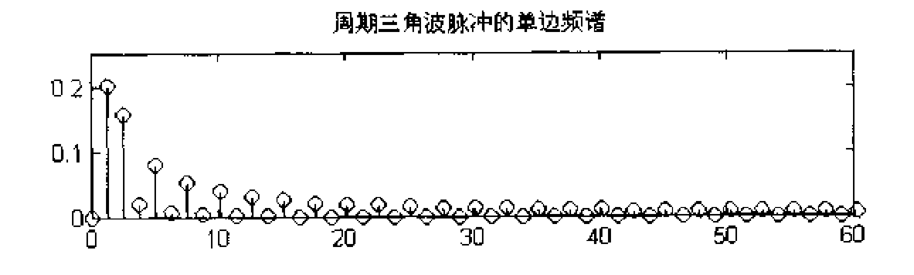


图 8-15　周期三角波脉冲单边频谱

8.3.3　用 FFT 实现周期信号的频谱分析

Fourier 分析在信号处理中占有重要地位，从时域角度来看，Fourier 级数可分为连续时间 Fourier 级数 CTFS 和离散时间 Fourier 级数 DTFS。快速 Fourier 算法的出现，使 Fourier 分析进入了新的应用阶段。本节将介绍如何利用快速 Fourier 变换 FFT（Fast Fourier Transform）对连续时间 Fourier 级数 CTFS 展开进行数值计算。

1．用 FFT 计算离散时间 Fourier 级数 DTFS

已知周期为 N 的离散序列 $x(n)$，它的 DTFS 综合和分析公式分别由式(8-12)和(8-13)给出：

$$\text{IDTFS:}\qquad x(n)=\sum_{k=0}^{N-1}a_k e^{j(2\pi/N)nk},\quad n=0,1,\cdots,N-1 \tag{8-12}$$

$$\text{DTFS:}\qquad a_k=\frac{1}{N}\sum_{n=0}^{N-1}x(n)e^{-j(2\pi/N)nk},\quad k=0,1,\cdots,N-1 \tag{8-13}$$

MATLAB 内有两个非常高效的函数用来计算式（8-12）IDTFS 和式（8-13）DTFS 的程序，它就是 FFT 和 IFFT 函数。

若已知周期为 N 的离散序列 x（n），$n=0,1,\cdots,N-1$，它的 IFFT 和 FFT 公式分别由式（8-14）和（8-15）给出：

$$\text{IFFT:}\qquad x(n)=\frac{1}{N}\sum_{k=0}^{N-1}a_k e^{j(2\pi/N)nk},\quad n=0,1,\cdots,N-1 \tag{8-14}$$

$$\text{FFT:}\qquad a_k=\sum_{n=0}^{N-1}x(n)e^{-j(2\pi/N)nk},\quad k=0,1,\cdots,N-1 \tag{8-15}$$

若已知周期为 N 的离散序列 x（n），$n=0,1,\cdots,N-1$，利用 IFFT 和 FFT 计算 IDTFS 和 DTFS 公式分别由式（8-16）和（8-17）给出：

$$\text{IDTFS:}\qquad a=\frac{1}{N}*fft(x) \tag{8-16}$$

$$\text{DTFS:}\qquad x=N*ifft(a) \tag{8-17}$$

式（8-16）中，a 即为（8-13）式中的 a_k，a 包含有 a_k 中的 $k=0,1,\cdots,N-1$ 的 N 点向量。同理，x 即为（8-12）式中的 $x(n)$，x 包含有 $x(n)$ 中的 $n=0,1,\cdots,N-1$ 的 N 点向量。

2．用 FFT 计算连续时间 Fourier 级数 CTFS

为利用 FFT 计算连续时间周期信号 Fourier 级数复指数形式展开式，即计算（8-8）式的 $F_n=\frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)e^{-jn\Omega t}\,\mathrm{d}t$，需对该式进行离散化，即取足够小的 $\Delta t=T/N$，使得在这小区间上的积分可用"矩形面积：$f(k\Delta t)e^{-jn\omega_0 k\Delta t}\Delta t$"近似，并考虑到 $\omega_0=2\pi f_0=\frac{2\pi}{T}$、$t=k\Delta t$，于是得：



$$F_n = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} f(k\Delta t)e^{-jn\omega_0 k\Delta t}\,\mathrm{d}t = \Delta t.\frac{1}{T}\sum_{n=0}^{N-1} f(k\Delta t)e^{-j(2\pi/N)nk}$$

$$= \frac{1}{N}\sum_{n=0}^{N-1} f(k\Delta t)e^{-j(2\pi/N)nk} \tag{8-18}$$

即用 FFT 计算连续时间周期信号 $f(t)$ 的 Fourier 级数复指数形式展开式（8-18），可写为下列（8-19）式：

$$F = \frac{1}{N} * fft(f) \tag{8-19}$$

式（8-19）中，F 即为（8-18）式中的 F_n，F 包含有 F_n 中的 $n = 0,1,\cdots,N-1$ 的 N 点向量。

下列程序中，为了区分信号 $f(t)$ 和信号的频率 f，用 x 或 y 表示信号 $f(t)$，用 f 表示信号的频率。用 X 表示信号 x 的 $fft(f)$ 的值，cn 表示（8-19）中 F 的值为 $\frac{X}{N}$。

3．用 FFT 实现周期信号的频谱分析

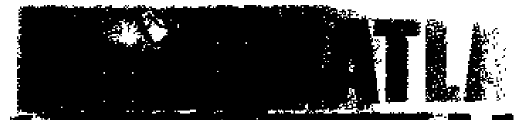
例 8-6：用 FFT 求图 8-1 所示的周期矩形脉冲函数的频谱，并绘出频谱图。其幅度为 1，脉冲宽度为 $tao = 1$（周期 T=5*tao）。

解：

1）编写函数文件 CTFSfft.m

编写函数文件 CTFSfft.m 如下所示：

```
%  [CTFSfft.m]     这是用 FFT 计算周期矩形脉冲双边频谱的程序
function [A,B,C,fn,t,x]=CTFSfft
% 利用 FFT，计算[0,T]区间上定义的时间波形的 Fourier 级数展开系数 A,B 和频谱 C,fn
%    T          时间波形周期
%    M          用做 2 的幂次
%    Nf         输出谐波的阶次，决定 A,B 的长度为(Nf+1)。Nf 不要超 2^(M-1)
%    A,B        分别是 Fourier 级数中 cos,sin 展开项的系数。A(1)是直流量
%    C          是定义在[-fs/2,fs/2]上的频谱
%    t,x        是原时间波形数据对
T=5;
if (nargin<2 | isempty(M));M=8;end
if nargin<3;Nf=input('plear Input 所需展开的最高谐波次数:Nf=');end
N=2^M;      %  使总采样点是 2 的整数倍
f=1/T;      % 被变换函数的频率
w0=2*pi*f;
dt=T/N;     % 时间分辨率
```




```
n=0:1:(N-1);  % 采样点序列
t=n*dt;         % 采样时间序列
x=time_fun(t,T);      % 被变换时间函数的采样序列, 调用 CTFSfft.m 的子函数
X=fft(x);      % 给出 n=0,1,…,N-1 上的 DFT 数据值
cn=X/N;          % 据式(8.19)计算 n=0,1,…,N-1 上的 FS 系数
z_cn=find(abs(cn)<1.0e-10);       % 寻找有限字长运算而产生(原应为 0)的"小"复数
cn(z_cn)=zeros(length(z_cn),1); % 强制那些"小"复数为 0
cn_SH=fftshift(cn);  % 计算 n=-N/2,…,-1,0,1,…,(N/2)-1 上的 FS 系数
C=[cn_SH cn_SH(1)];   %  形成关于 0 对称的（N+1）个 FS 系数
A(1)=C(N/2+1);
A(2:N/2+1)=2*real(C((N/2+2):end));
B(2:N/2+1)=-2*imag(C((N/2+2):end));
if Nf>N/2;error([' Nf 应小于 ' int2str(N/2-1)]);end
A(Nf+2:end)=[];
B(Nf+2:end)=[];
n1=-N/2*2*pi/T:2*pi/T:N/2*2*pi/T;
                      % 产生总点数为（N+1）个关于 0 对称的 FS 系数
fn=n1*f;              % 关于 0 对称的频率分度序列
y=time_fun_e       % 调用 CTFSfft.m 的子函数
subplot(2,1,2),
stem(n1,abs(C));
title('周期矩形脉冲频谱')
axis([-150,150,0,0.12])
hold off
%---
function     x=time_fun(t,T)
% 该函数是 CTFSfft.m 的子函数。
%    t            是时间数组
%    T            是周期
A=1;tao=1;T=5;
x=zeros(size(t));ii=find(t>=-tao /2& t<=tao/2);
x(ii)=ones(size(ii)).*A;x(t==0)=1;
%---
function y=time_fun_e
% 该函数是 CTFSfft.m 的子函数。
%    t            是时间数组
%    T            是周期     duty=tao/T=0.2
T=5;t=-2*T:0.01:2*T;tao=T/5;
y=rectpuls(t,1);          % 产生一个宽度 tao=1 的矩形脉冲
```



```
subplot(2,2,2)
plot(t,y)
hold on
y=rectpuls(t-5,1);    % 产生一个宽度 tao=1 的矩形脉冲,中心位置在 t=5 处
plot(t,y)
hold on
y=rectpuls(t+5,1);    % 产生一个宽度 tao=1 的矩形脉冲,中心位置在 t=-5 处
plot(t,y)
title('周期为 T=5，脉宽 tao=1 的矩形脉冲')
axis([-10,10,0,1.2])
```

2）程序运行结果

调用函数 CTFSfft.m，即可绘出周期矩形脉冲的频谱，如图 8-16 所示。

指令如下：在 MATLAB 命令窗口键入 CTFSfft，并按回车键，命令窗口将出现 pleas Input 所需展开的最高谐波次数 *Nf*=？，本例输入 *Nf*=60，即可绘出周期矩形脉冲的频谱图，如图 8-16 所示。

将图 8-16 与图 8-7 的 CTFS 分析结果比较可知，用 FFT 分析连续时间周期信号频谱与用 Fourier 级数分析连续时间周期信号的频谱结果一致。

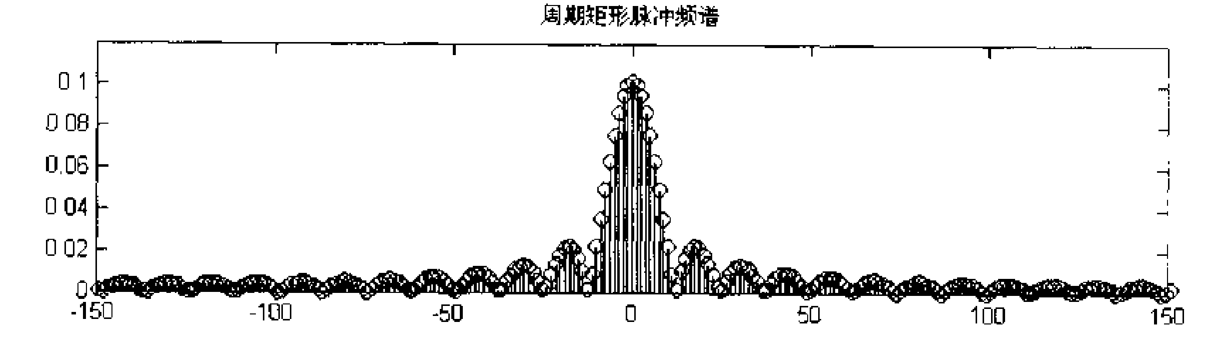


图 8-16　用 FFT 分析周期矩形脉冲及其频谱

上机练习题

1．定义符号函数 $sign(n)=\begin{cases}1 & n>0\\ 0 & n=0\\ -1 & n<0\end{cases}$

考虑具有下面基波周期 T 和指数形式的傅里叶级数系数 $\{F_n\}$ 的信号。

（1）做出三角形式或指数形式的频谱图。（要求谐波次数>10）

（2）分别绘出基波；基波+三次谐波；基波+三次谐波+五次谐波；基波+三次谐波+五次谐波+七次谐波的信号合成的波形。

（a） $x_1(t)$：

$$T=5;\quad F_n=\begin{cases}\dfrac{1}{(n^2+1)}, & |n|\leqslant 10\\ 0, & |n|>10\end{cases}$$

（b） $x_2(t)$：

$$T=20;\quad F_n=\begin{cases}sign(n)\dfrac{j}{(-2)^{|n|}}, & |n|\leqslant 10\\ 0, & |n|>10\end{cases}$$

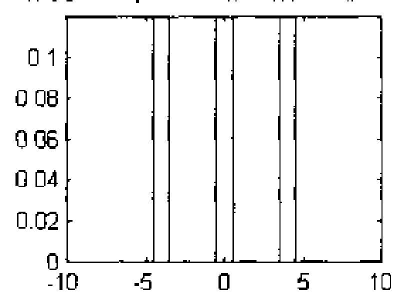
（c） $x_3(t)$：

$$T=5;\quad a_n=\begin{cases}\dfrac{1}{2^{|n+2|}}, & |n|\leqslant 10\\ 0, & |n|>10\end{cases}$$

2．已知周期矩形脉冲信号如下图所示，其基波周期 T=4，脉宽 *tao*=1。在-1<t<1 区间内该矩形波由下式表示：

$$x(t)=\begin{cases}1, & |t|\leqslant\dfrac{1}{2}\\ 0, & |t|\geqslant\dfrac{1}{2}\end{cases}$$

试用 MATLAB 求该信号三角形式的傅里叶系数 a_n，并绘出它的频谱图。



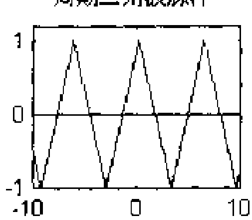
（周期 T=4，脉宽 *tao*=1 的周期矩形波）

3．已知基波周期 T=4 的三角波如下图所示，即在第一个周期（-1<t<1）内信号可表示为：

$$x(t)=1-|t|$$

试用 MATLAB 求该信号的傅里叶级数，绘出它的频谱图。将它的频谱图与方波的频谱图做比较。

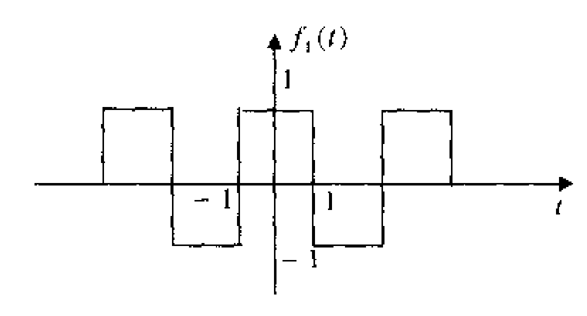


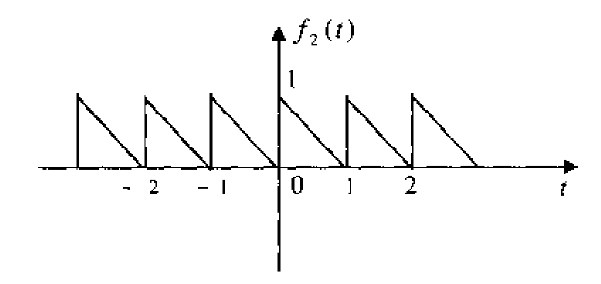


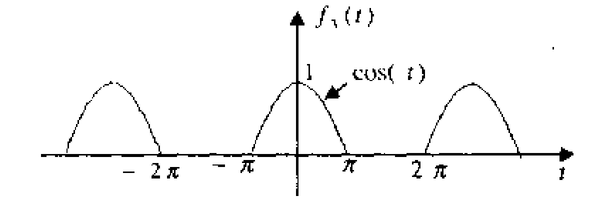
（周期 T=5 的标准三角波）

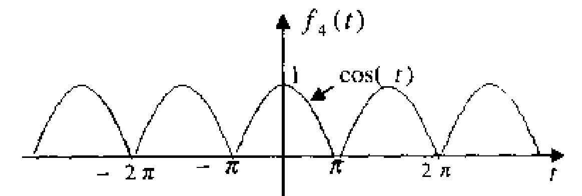
4．已知周期信号的波形分别如下面的四张图，试用 MATLAB 编程求出它们的傅里叶系数，绘出其直流、一次、二次、三次、四次及五次谐波叠加后的波形图，并将其与原周期信号的时域波形进行比较，观察周期信号的分解与合成过程。

5．对下面的四张图所示的周期信号，试用 MATLAB 编程绘出信号的振幅频谱，改变信号的周期（增大或减小），再绘出其频谱图，观察信号周期与频谱的关系，当周期 t 趋于无穷大时，频谱谱线将发生什么样的变化？







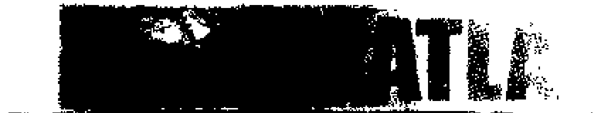




日期：________ 进程：________ 备注：________

第9章 连续时间信号的频域分析及MATLAB实现





9.1 傅里叶变换及 MATLAB 实现

信号 $f(t)$ 的傅里叶变换定义为：

$$F(j\omega)=\int_{-\infty}^{\infty}f(t)e^{-j\omega t}\,\mathrm{d}t \tag{9-1}$$

值得注意的是，$f(t)$ 的傅里叶变换存在的充分条件是 $f(t)$ 在无限区间内绝对值可积，即 $f(t)$ 满足下式：

$$\int_{-\infty}^{\infty}|f(t)|\,\mathrm{d}t<\infty \tag{9-2}$$

但式（9-2）并非 $f(t)$ 存在的必要条件。当引入 $f(t)$ 的广义函数概念后，使一些不满足绝对可积的 $f(t)$ 也能进行傅里叶变换。

傅里叶逆变换的定义是：

$$f(t)=\int_{-\infty}^{\infty}F(j\omega)e^{j\omega t}\,\mathrm{d}\omega \tag{9-3}$$

MATLAB 的 Symbolic Math Toolbox 提供了能直接求解傅里叶变换及逆变换的函数 fourier()及 ifourier()。两者的调用格式如下。

1．Fourier 变换

（1）F=fourier(f)

（2）F=fourier(f,v)

（3）F=fourier(f,u,v)

说明

（1）F=fourier(f)是符号函数 f 的 Fourier 变换，默认返回是关于 ω 的函数。如果 $f=f(\omega)$，则 fourier 函数返回关于 t 的函数。

（2）F=fourier(f,v)返回函数 F 是关于符号对象 v 的函数，而不是默认的 ω，即：

$$F(v)=\int_{-\infty}^{\infty}f(x)e^{-jvx}\,\mathrm{d}x$$

（3）F=fourier(f,u,v)对关于 u 的函数 f 进行变换，返回函数 F 是关于 v 的函数。即：

$$F(v)=\int_{-\infty}^{\infty}f(u)e^{-jvu}\,\mathrm{d}u$$

2．Fourier 逆变换

（1）f=ifourier(F)



（2）f-ifourier(F,u)

（3）f=ifourier(F,v,u)

说明

（1）f=ifourier(F)是函数 F 的 Fourier 逆变换，默认的独立变量为 ω，默认返回是关于 x 的函数。如果 $F=F(x)$，则 ifourier 返回关于 t 的函数。

（2）f=ifourier(F,u)返回函数 f 是 u 的函数，而不是默认的 x 的函数。

（3）f=ifourier(F,v,u)对关于 v 的函数 F 进行变换，返回关于 u 的函数 f。

注释

在调用函数 fourier()及 ifourier()之前，要用 syms 命令对所用到的变量（如 t、u、v、w）等进行说明，即要将这些变量说明成符号变量。对 fourier()中的函数 f 及 ifourier()的函数 F，也要用符号定义符 sym 将 f 或 F 说明为符号表达式；若 f 或 F 是 MATLAB 中的通用函数表达式，则不必用 sym 加以说明。以下举例说明如何调用该函数来实现傅里叶变换。

例 9-1：求 $f(t)=e^{-2|t|}$ 的傅里叶变换。

解：

我们可以用 MATLAB 解决上述问题，命令如下：

```
syms t
fourier(exp(-2*abs(t)))
ans=
4/(4+w^2)
```

若傅里叶变换的结果变量我们希望是 v，则可执行如下命令：

```
syms t v
fourier(exp(-2*abs(t)),t,v)
ans=
4/(4+v^2)
```

例 9-2:! 求 $F(j\omega)=\dfrac{1}{1+\omega^2}$ 的傅里叶逆变换 $f(t)$。

解：

实现该过程的 MATLAB 命令为：

```
syms t w
ifourier(1/(1+w^2),t)
ans=
1/2*exp(-t)*Heaviside(t)+1/2*exp(t)*Heaviside(-t)
```

其中，Heaviside(t)函数即为单位阶跃函数 $\varepsilon(t)$。

例 9-3：设 $f(t)=\dfrac{1}{2}e^{-2t}\varepsilon(t)$，试画出 $f(t)$ 及其幅频图。

解：

这里采用符号变量定义符 sym，并用符号函数作图函数 ezplot()绘图。实现该过程的命令文件如下：

```
syms t v w x;              %生成符号表达式;
x=1/2*exp(-2*t)*sym('Heaviside(t)');
F=fourier(x);
subplot(211);
ezplot(x);
subplot(212);
ezplot(abs(F));
```

程序绘制的信号波形及频幅图如图9-1所示。

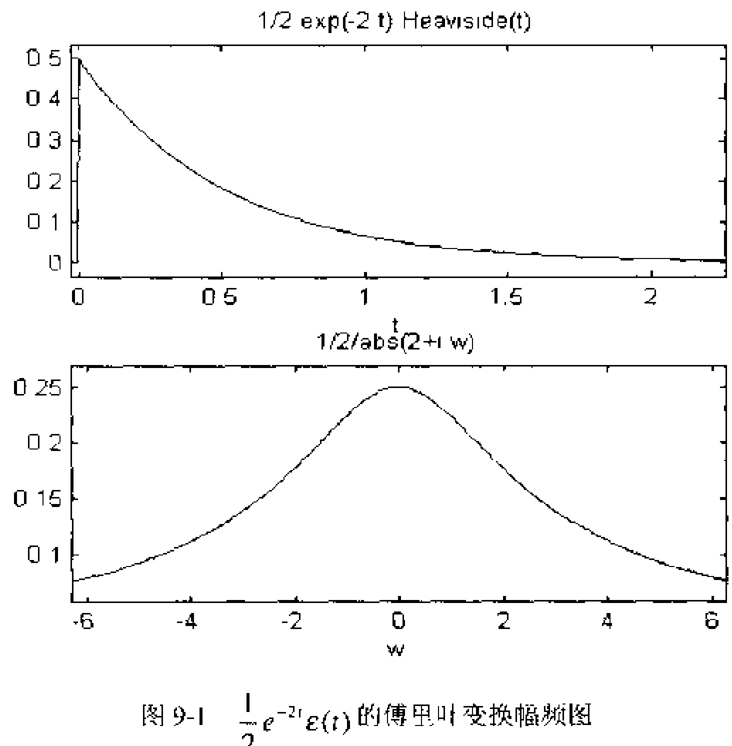


图9-1 $\frac{1}{2}e^{-2t}\varepsilon(t)$的傅里叶变换幅频图

程序中的 Heaviside(t)是调用了 Symbolic Math Toolbox 的 Heaviside.m 文件，内容为：

```
function f=Heaviside(t)
 f=(t>0);
```

注意

- 调用 Heaviside(t)时，最好检查一下您的工作目录下的 Heaviside.m 文件中的 f=(t>0)中 t 的定义范围。若有的为 f=(t>2)，这时要改过来。
- 要利用 sym 来创建符号表达式 x。

采用 fourier()及 ifourier()得到的返回函数，仍然是符号表达式。若需对返回函数作图，则应用 ezplot()绘图命令而不能用 plot()命令。如果返回函数中含有诸如狄拉克函数 δ(ω)等的项，则用 ezplot()也无法作图。此外，用 fourier()对某些信号求变换时，其返回函数可能会包含一些不能直接表达的式子，甚至可能会出现一些屏幕提示为“未被定义的函数或



变量"的项，此时更不用说对此返回函数作图了（参见上机练习题四的第 3 题）。这是 fourier() 的一个局限。另一个局限是在很多应用场合，原信号 $f(t)$ 尽管是连续的，但却不可能表示成符号表达式，而更多的实际测量现场获得的信号是多组离散的数值量 $f(k)$，此时也不可能应用 fourier() 对 $f(k)$ 进行处理，而只能应用 9.2 节介绍的方法来求信号的傅里叶变换。

9.2　连续时间信号傅里叶变换的数值计算

严格说来，若不使用 symbolic 工具箱，是不能分析连续信号的。为了能更好地体会 MATLAB 的数值计算功能，特别是其强大的矩阵运算能力，这里给出连续信号傅氏变换的数值计算方法。先给出算法的理论依据如下：

$$F(j\omega)=\int_{-\infty}^{\infty}f(t)e^{-j\omega t}\,\mathrm{d}t=\lim_{\tau\to 0}\sum_{n=-\infty}^{n=\infty}f(n\tau)e^{-j\omega n\tau}\tau \tag{9-4}$$

对于一大类信号，当取 τ 足够小时，上式的近似情况可以满足实际需要。若信号 $f(t)$ 是时限的，或当 $|t|$ 大于某个给定值时，$f(t)$ 的值已衰减得很厉害，可以近似地看成时限信号时，则式（9-4）中的 n 取值就是有限的，设为 N，有：

$$F(k)=\tau\sum_{n=0}^{N-1}f(n\tau)e^{-j\omega_k n\tau}\;,\;\;0\leqslant k\leqslant N \tag{9-5}$$

上式是对式（9-4）中的频率 ω 进行取样，通常：

$$\omega_k=\frac{2\pi}{N\tau}k \tag{9-6}$$

采用 MATLAB 实现（9-5）式时，其要点是要正确生成 $f(t)$ 的 N 个样本 $f(n\tau)$ 的向量 f 及向量 $e^{-j\omega_k n\tau}$，两向量的内积（即两矩阵的乘积）结果即完成式（9-5）的计算。

此外，还应注意时间取样间隔 τ 的确定。其依据是 τ 需小于奈奎斯特（Nyquist）取样间隔。如果对于某个信号 $f(t)$，它不是严格的带限信号，则可根据实际计算的精度要求来确定一个适当的频率 ω_0 为信号的带宽。下面举例说明。

例 9-4：已知门信号 $f(t)=g_2(t)=\begin{cases}1 & |t|<1\\ 0 & |t|>1\end{cases}$，求其傅里叶变换 $F(j\omega)$。

解：

$f(t)$ 可用两个阶跃信号来表示，即：

$$f(t)=\varepsilon(t+1)-\varepsilon(t-1)$$

由信号分析可知，该信号的频谱为 $F(j\omega)=2Sa(\omega)$，其第一个过零点频率为 π，一般将此频率认为是信号 $f(t)$ 的带宽。考虑到的 $F(j\omega)$ 形状，将精度提高到该值的 50 倍，即 $\omega_0=50\omega_1=50\pi$，据此确定取样间隔：



$$\tau < \frac{1}{2F_0} = \frac{1}{2\times\frac{\omega_0}{2\pi}} = 0.02$$

实现该过程的 MATLAB 命令程序如下：

```
R=0.02;t=-2:R:2;
f=Heaviside(t+1)-Heaviside(t-1);
W1=2*pi*5;                              %频率宽度
N=500;k=0:N;W=k*W1/N;                  %采样数为 N,W 为频率正半轴的采样点
F=f*exp(-j*t'*W)*R;                  %求 F(jw)
F=real(F);
W=[-fliplr(W),W(2:501)];              %形成负半轴及正半轴的 2N+1 个频率点 W
F=[fliplr(F),F(2:501)];             %形成对应于 W 的 F(jw)的值
subplot(2,1,1);plot(t,f);
xlabel('t');ylabel('f(t)');
title('f(t)=u(t+1)-u(t-1)');
subplot(2,1,2);plot(W,F);
xlabel('w');ylabel('F(w)');
title('f(t)的傅氏变换 F(w)');
```

程序执行出现如图 9-2 所示的曲线。显然，该曲线与我们熟知的理论结果完全吻合。

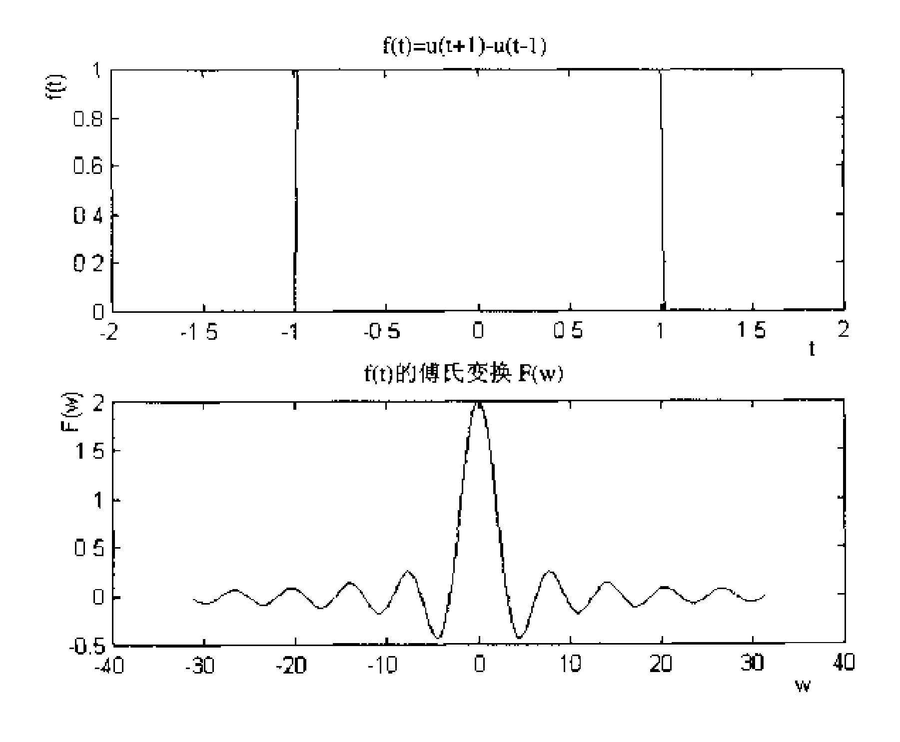


图 9-2　门信号时域波形及频谱曲线



9.3 信号的幅度调制及 MATLAB 实现

设信号 $f(t)$ 的频谱为 $F(j\omega)$，现将 $f(t)$ 乘以载波信号 $\cos(\omega_0 t)$，得到高频的已调信号 $y(t)$，即：

$$y(t)=f(t)\cos(\omega_0 t) \tag{9-7}$$

$f(t)$ 称为调制信号。实现信号调制的原理图如图 9-3 所示。

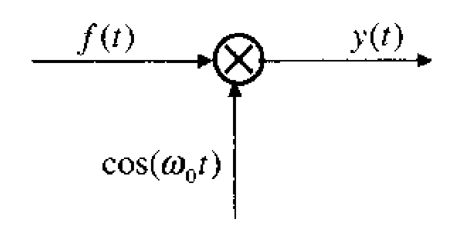


图 9-3　幅度调制原理图

从频域上看，已调制信号 $y(t)$ 的频谱为原调制信号 $f(t)$ 的频谱搬移到 $\pm\omega_0$ 处，幅度降为原 $F(j\omega)$ 的 1/2，即：

$$f(t)\cos(\omega_0 t)\leftrightarrow\frac{1}{2}\{F[j(\omega+\omega_0)]+F[j(\omega-\omega_0)]\} \tag{9-8}$$

上式即为调制定理，也是傅里叶变换性质中“频移特性”的一种特别情形。

说明

这里采用的调制方法为抑制载波方式，即 $y(t)$ 的频谱中不含有 $\cos(\omega_0 t)$ 的频率分量。

MATLAB 提供了专门的函数 modulate（）用于实现信号的调制。调用格式为：

```
y=modulate(x,Fc,Fs,'method')
[y,t]=modulate(x,Fc,Fs)
```

其中，*x* 为被调信号，*Fc* 为载波频率，*Fs* 为信号 *x* 的采样频率，method 为所采用的调制方式，若采用幅度调制、双边带调制、抑制载波调制，则'method'为'am'或'amdsd-sc'。其执行算法为：

```
y=x*cos(2*pi*Fc*t)
```

其中 *y* 为已调制信号，*t* 为函数计算时间间隔向量。

下面举例说明如何调用函数 modulate（）来实现信号的调制。

例 9-5：设信号 $f(t)=\sin(100\pi t)$，载波 $y(t)$ 为频率为 400Hz 的余弦信号。试用 MATLAB 实现调幅信号 $y(t)$，并观察 $f(t)$ 的频谱和 $y(t)$ 的频谱，以及两者在频域上的关系。

解：

在下面的 MATLAB 的实现的程序中，为了观察 $f(t)$ 及 $y(t)$ 的频谱，在这里介绍一个 MATLAB 的“信号处理工具箱函数”中的估计信号的功率谱密度函数 psd()，其格式是：

```
[Px, f]=psd(x, Nfft, Fs, window, noverlap, dflag)
```

其中，x 是被调制信号（即本例中的 $f(t)$），*Nfft* 指定快速傅氏变换 FFT 的长度，*Fs* 为对信号 x 的采样频率。后面三个参数的意义涉及到信号处理的更深的知识，在此暂不介绍。

用 MATLAB 完成本例的程序如下：

```
Fs=1000;                    %被调信号 x 的采样频率
Fc=400;                     %载波信号的载波频率
N=1000;                     %FFT 的长度
n=0:N-2;
t=n/Fs;
x=sin(2*pi*50*t);     %被调信号
subplot(221)
plot(t,x);
xlabel('t(s)');
ylabel('x');
title('被调信号');
axis([0 0.1 -1 1])
Nfft=1024;
window=hamming(512);
noverlap=256;
dflag='none';
[Pxx,f]=psd(x,Nfft,Fs,window,noverlap,dflag);      %求被调信号 x 的功率谱
subplot(222)
plot(f,Pxx)
xlabel('f(Hz)');
ylabel('功率谱(X)');
title('被调信号的功率谱')
grid
y=modulate(x,Fc,Fs,'am');                                %已调信号
subplot(223)
plot(t,y)
xlabel('t(s)');
ylabel('y');
axis([0 0.1 -1 1])
title('已调信号')
[Pxx,f]=psd(y,1024,Fs,window,noverlap,dflag);       %求已调信号 y 的功率谱
subplot(224)
plot(f,Pxx)
xlabel('f(Hz)');
ylabel('功率谱(Y)');
```

```
title('已调信号的功率谱');
grid
```

上述程序的运行结果如图 9-4 所示，其中左边上下两图为 $f(t)$ 及 $y(t)$ 信号，即时域波形，右边上下两图分别为对应 $f(t)$ 及 $y(t)$ 的功率谱。

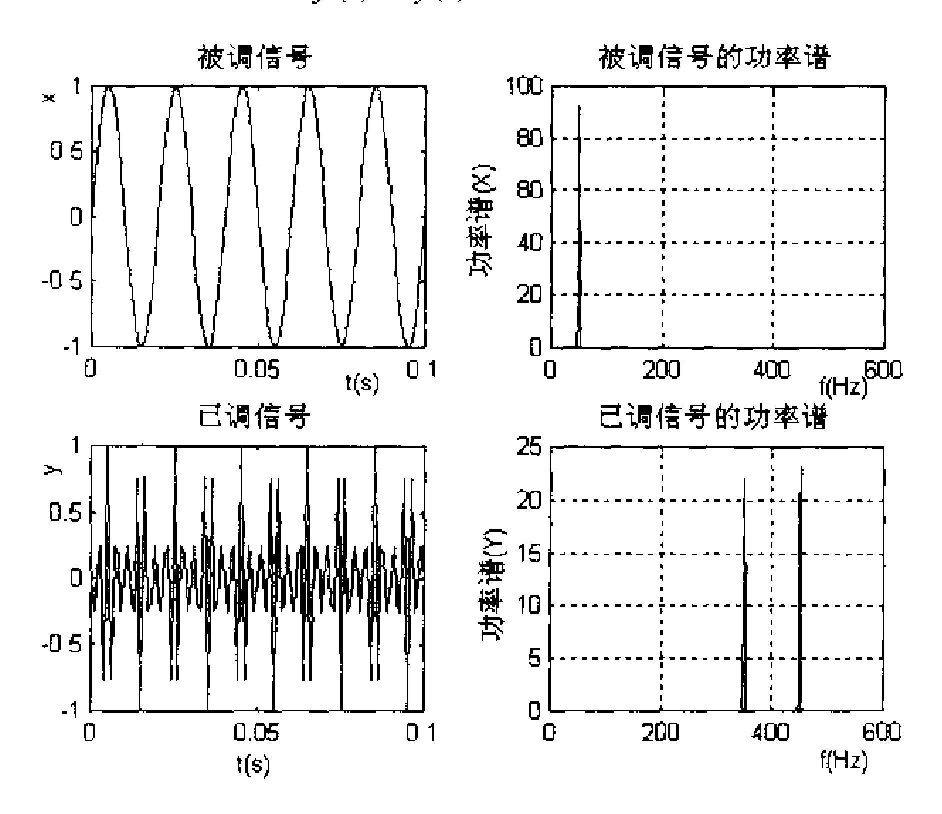


图 9-4　被调信号、已调信号及其谱线

由图 9-4 可见，$f(t)$ 的功率频谱处在频域的频率 f=400Hz 为中心的两侧、偏移值为 50Hz 的双边带。显然，上述结果与信号与系统分析的理论结果完全一致。

需要指出的是，一个信号的频谱与功率谱在数值上及定义上是有差别的，但两者的联系也是很密切的，其关系为：

$$p(\omega)=\lim_{T\to\infty}\frac{|F(j\omega)|^2}{T} \tag{9-9}$$

其中 T 为信号的周期。

本例中的主要目的是观察被调用信号 $f(t)$ 及已调用信号 $y(t)$ 的谱线在频域上的位置变化及关系，验证调制定理，而在数值上的差别予以忽略。另外，一般“信号与系统”教材介绍的信号调制多为幅度、双边带且抑制载波调制方式，所以例 9-5 也仅涉及这种方式。但是，函数 modulate（）中的“method”可设置多种调制方式以适合于具体的调制要求，有兴趣的读者可查阅相关资料，这里从略。

此外，也可以直接生成调制信号 $f_1(t)=f(t)\cos(\omega_0 t)$，并用 MATLAB 编程求 $f_1(t)$ 的频谱。用下例说明。

例 9-6：设 $f(t)=\varepsilon(t+1)-\varepsilon(t-1)$，$f_1(t)=f(t)\cos(10\pi t)$，试用 MATLAB 画出

$f(t)$、$f_1(t)$ 的时域波形及其频谱，并观察傅里叶变换的频移特性。

解：

实现该过程的 MATLAB 命令程序如下：

```
R=0.005;t=-1.2:R:1.2;
f=Heaviside(t+1)-Heaviside(t-1);
f1=f.*cos(10*pi*t);      %已调信号
subplot(221)
plot(t,f)
xlabel('t');
ylabel('f(t)');
subplot(222);
plot(t,f1);
xlabel('t');
ylabel('f1(t)=f(t)*cos(10*pi*t)');
W1=40;
N=1000;
k=-N:N;
W=k*W1/N;
F=f*exp(-j*t'*W)*R;                        求 F(jw)
F=real(F);
F1=f1*exp(-j*t'*W)*R;                      求 F1(jw)
F1=real(F1);
subplot(223);
plot(W,F);
xlabel('w');
ylabel('F(jw)');
subplot(224);
plot(W,F1);
xlabel('w');
ylabel('F1(jw)');
```

程序运行结果如图 9-5 所示，

由图 9-5 可见，$f_1(t)$ 的频谱 $F_1(j\omega)$ 即是将 $f(t)$ 的频谱 $F(j\omega)$ 搬移到 $\pm10\pi$ 处，且幅度为 $F(j\omega)$ 的幅度的一半。



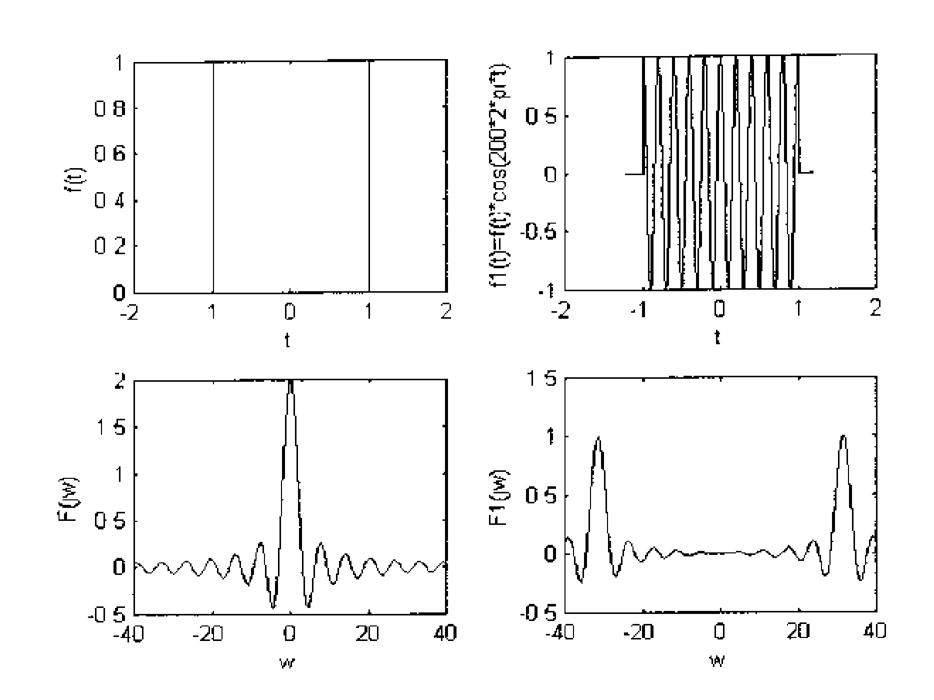


图 9-5　原信号 $f(t)$、调制信号 $f_1(t)$ 的波形及其频谱 $F(j\omega)$、$F_1(j\omega)$

9.4　傅里叶变换的性质及 MATLAB 实现

傅里叶变换的性质包括线性、奇偶性、对称性、尺度变换、时移、频移、卷积定理（频域及时域）、微分性（频域及时域）、积分性（频域及时域）等。在这里。我们通过具体的例子，分别给出傅里叶变换中的尺度变换特性、时移特性、频移特性、时域卷积定理、对称性及时域微分性的 MATLAB 实现。

9.4.1　傅里叶变换的尺度变换特性

若 $f(t) \leftrightarrow F(j\omega)$，则傅里叶变换的尺度变换特性为：

$$f(at) \leftrightarrow \frac{1}{|a|}F(j\frac{\omega}{a}), \quad a \neq 0 \tag{9-10}$$

下面举例说明傅里叶变换的尺度特性。

例 9-7：设 $f(t) = \varepsilon(t+1) - \varepsilon(t-1) = g_2(t)$，即门宽为 τ=2 的门信号，用 MATLAB 求 $y(t) = \varepsilon(2t+1) - \varepsilon(2t-1) = g_1(t)$ 的频谱 $Y(j\omega)$，并与 $f(t)$ 的频谱 $F(j\omega)$ 进行比较。

解：

本题中，$y(t)$ 信号相当于原信号 $f(t)$ 在时域上压缩一倍，即 $y(t) = f(2t)$，$a = 2$，按（9-10）式，$Y(j\omega)$ 的频域宽度应是 $F(j\omega)$ 的两倍，而幅度下降为 $F(j\omega)$ 的一半。

$f(t)$ 的频谱 $F(j\omega)$ 已在例 9-4 中给出。在该例的 MATLAB 程序中，将信号改为：f = Heaviside(2*t+1) - Heaviside(2*t-1)，其他语句不变。这样形成的程序即为本例的 MATLAB 程序。程序运行的结果如图 9-6 所示。为便于观察比较，请读者将图 9-2 与图 9-6 对比起来看，显然，$Y(j\omega)$ 将 $F(j\omega)$ 展宽了一倍，而幅度降为 $F(j\omega)$ 的幅度的一半。

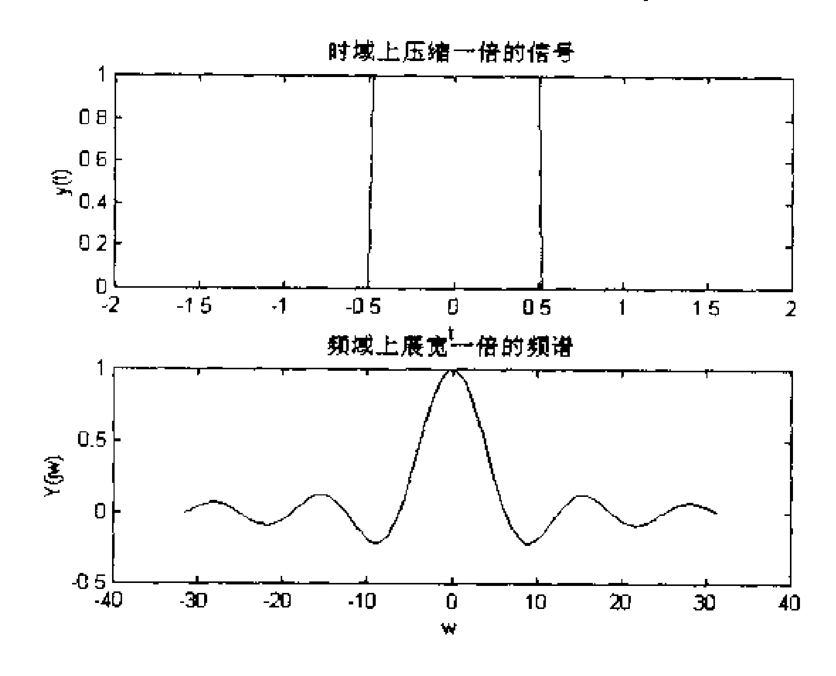


图 9-6 尺度变换的例子

9.4.2 傅里叶变换的时移特性

若 $f(t) \leftrightarrow F(j\omega)$，则傅里叶变换的时移特性为：

$$f(t \pm t_0) \leftrightarrow f(j\omega)e^{\pm j\omega t_0} \tag{9-11}$$

下面举例说明傅里叶变换的时移特性。

例 9-8：设 $f(t) = \frac{1}{2}e^{-2t}\varepsilon(t)$，试用 MATLAB 绘出 $f(t)$ 及其频谱（幅度谱及相位谱）。

解：

程序为下列命令文件：

```
r=0.02;
t=-5:r:5;
N=200;
W=2*pi*1;
k=-N:N;
w=k*W/N;
f1=1/2*exp(-2*t).*Heaviside(t);                %定义 f(t)
F=r*f1*exp(-j*t'*w);                           %求 f(t)的傅里叶变换 F(jω)
```

```
F1=abs(F);                                  %求F(jω)的幅度
P1=angle(F);                                % 求F(jω)的相位
subplot(311);
plot(t,f1);
grid;
xlabel('t');
ylabel('f(t)');
title('f(t)');
subplot(312);
plot(w,F1);
xlabel('w');
grid;
ylabel('F(jw)');
subplot(313);
Plot(w,P1*180/pi);
grid;
xlabel('w');
ylabel('P(度)');
```

程序运行结果如图9-7所示。

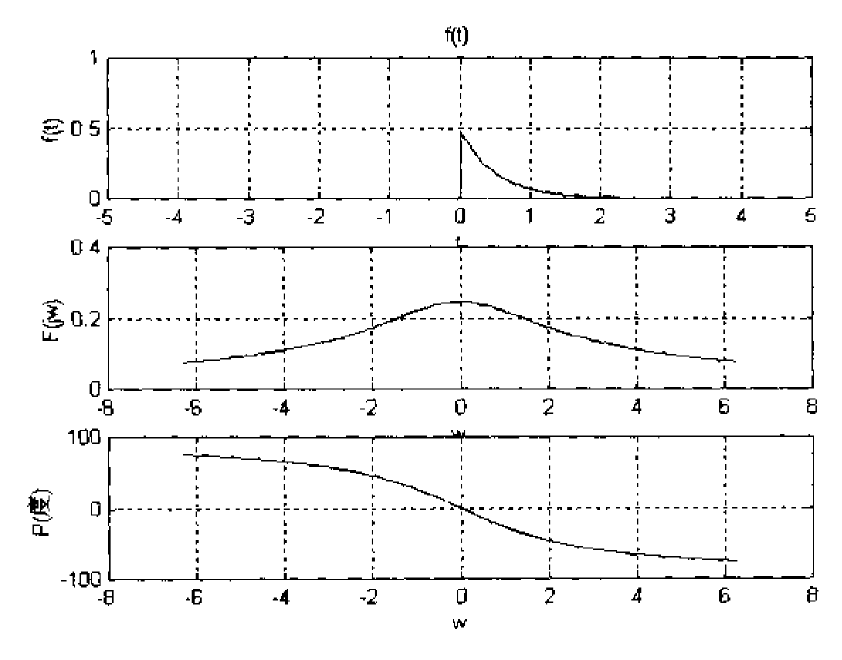


图 9-7　$f(t)$及其幅频特性与相频特性

例 9-9：设 $f(t)=\frac{1}{2}e^{-2(t-0.3)}\varepsilon(t-0.3)$，求用 MATLAB 绘出信号 $f(t)$ 及其频谱，观察信号时移对信号频谱的影响。

解：

MATLAB 实现的程序为下列命令文件：

```
r=0.02;
t=-2:r:2;
N=200;
W=2*pi*1;
k=-N:N;
w=k*W/N;
f1=1/2*exp(-2*(t-0.3)).*Heaviside(t-0.3);          %定义 f(t)
F=r*f1*exp(-j*t'*w);                                %求 f(t)的傅里叶变换 F(jω)
F1=abs(F);                                          %求 F(jω)的幅度
P1=angle(F);                                        %求 F(jω)的相位
subplot(311);
plot(t,f1);grid;
xlabel('t');ylabel('f(t)');
title('f(t)');
subplot(312);
plot(w,F1);grid;
xlabel('w');ylabel('幅度');
subplot(313);
plot(w,P1*180/pi);grid;
xlabel('w');ylabel('相位(度)');
```

程序运行结果如图9-8所示。由图可见，与例9-8的图9-7相比，可知当时域波形右移后幅度谱不变，相位增加-0.3ω。

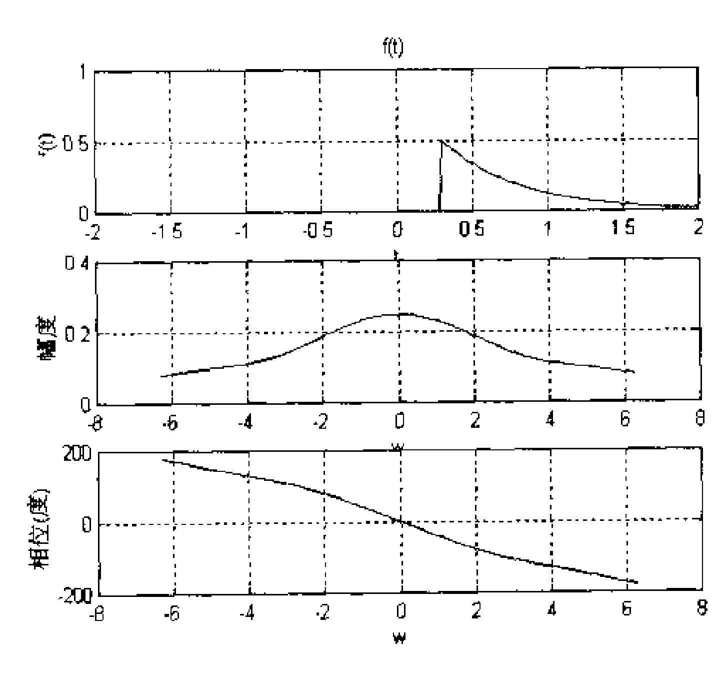


图 9-8　$f(t)$ 及其幅频特性与相频特件



例 9-10：$f(t)=\frac{1}{2}e^{-2(t+0.3)}\varepsilon(t+0.3)$，求 $f(t)$ 的频谱。

解：

对例 9-9 的程序稍加改动，即可完成。程序运行结果如图 9-9 所示。

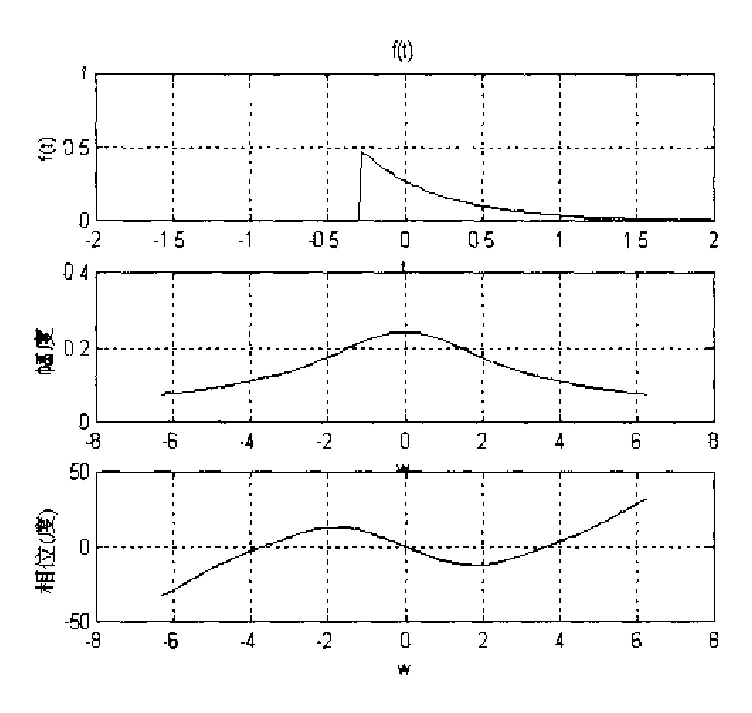


图 9-9 $f(t)$及其幅频特性与相频特性

从图 9-9 我们可以看出，信号时移后其振幅频谱并没有发生改变，而只是相位频谱发生了变化（增加 0.3ω）。

9.4.3 傅里叶变换的频移特性

若 $f(t)\leftrightarrow F(j\omega)$，则傅里叶变换的频移特性为：

$$f(t)e^{\pm j\omega_0 t}\leftrightarrow F[j(\omega\mp\omega_0)] \tag{9-12}$$

例 9-11：设 $f(t)=\varepsilon(t+1)-\varepsilon(t-1)$，试用 MATLAB 绘出 $f_1(t)=f(t)e^{-j20t}$ 及 $f_2(t)=f(t)e^{j20t}$ 的频谱 $F_1(j\omega)$及$F_2(j\omega)$，并与 $f(t)$ 的频谱 $F(j\omega)$ 进行比较。

解：

用 MATLAB 实现的程序如下：

```
R=0.02;t=-2:R:2;
f=Heaviside(t+1)-Heaviside(t-1);
f1=f.*exp(-j*20*t);
f2=f.*exp(j*20*t);
W1=2*pi*5;
N=500;k=-N:N;W=k*W1/N;
```

```
F1=f1*exp(-j*t'*W)*R;                    %求f1(t)的傅里叶变换F1(jω)
F2=f2*exp(-j*t'*W)*R;                    %求f2(t)的傅里叶变换F2(jω)
F1=real(F1);
F2=real(F2);
subplot(121);
plot(W,F1);
xlabel('w');
ylabel('F1(jw)');
title('F(w)左移到 w=20 处的频谱 F1(jw)');
subplot(122);
plot(W,F2);
xlabel('w');
ylabel('F2(jw)');
title('F(w)右移到 w=20 处的频谱 F2(jw)');
```

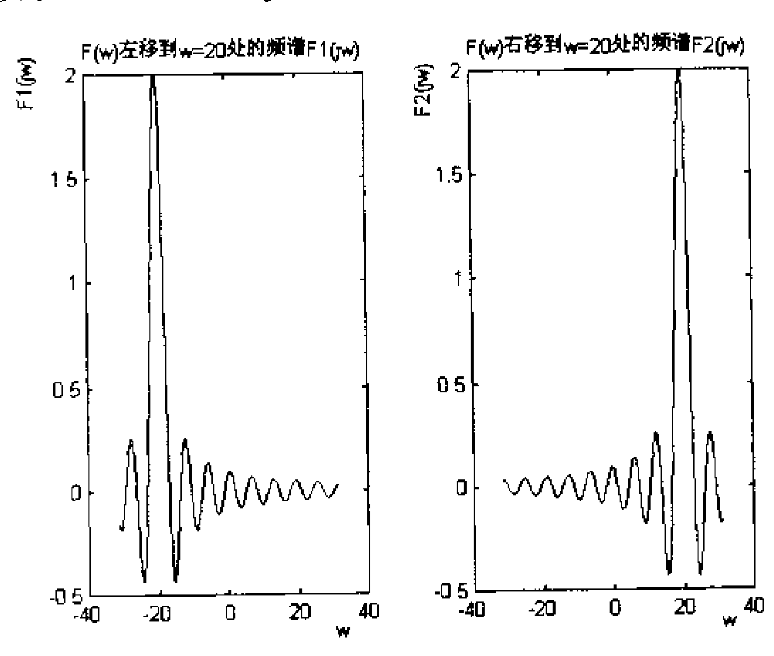


图 9-10 傅里叶变换的频移特性的例子

由图 9-10 可见，对比例 9-4 的结果可知 $F_1(j\omega)$ 及 $F_2(j\omega)$ 是将 $F(j\omega)$ 分别搬移到 $\omega=-20$ 及 $\omega=20$ 处的频谱。

9.4.4 傅里叶变换的时域卷积定理

变换的时域卷积定理如下。

若信号 $f_1(t), f_2(t)$ 的傅里叶变换分别为 $F_1(j\omega), F_2(j\omega)$，则：

$$f_1(t) * f_2(t) \leftrightarrow F_1(j\omega)\cdot F_2(j\omega) \tag{9-13}$$



例 9-12：设 $f(t)=\varepsilon(t+1)-\varepsilon(t-1)$，$y(t)=f(t)*f(t)$，试用 MATLAB 给出 $f(t)$、$y(t)$、$F(j\omega)$、$F(j\omega)\cdot F(j\omega)$及 $Y(j\omega)$的图形，验证式（9-13）的时域卷积定理。

解：

MATLAB程序如下：

```
R=0.05;t=-2:R:2;
f=Heaviside(t+1)-Heaviside(t-1); %f(t)的时域宽度为 2，t 的取值范围放大为-2～2
subplot(321)
plot(t,f)
xlabel('t');
ylabel('f(t)');
y=R*conv(f,f);                    %求 y(t)=f(t)*f(t),本例 y(t)的时宽为 f(t)的时宽两倍
n=-4:R:4;                          %n 的取值范围为 t 的取值范围的两倍，为-4～+4
subplot(322);
plot(n,y);
xlabel('t');
ylabel('y(t)=f(t)*f(t)');
axis([-3 3 -1 3]);
W1=2*pi*5;
N=200;
k=-N:N;
W=k*W1/N;
F=f*exp(-j*t'*W)*R;                %求 f(t)的傅里叶变换 F(jω)
F=real(F);
Y=y*exp(-j*n'*W)*R;              %求 y(t)的傅里叶变换 Y(jω)
Y=real(Y);
F1=F.*F                                %求 F(jω)×F(jω)
subplot(323);
plot(W,F);
xlabel('w');
ylabel('F(jw)');
subplot(324);
plot(W,F1);
xlabel('w');
ylabel('F(jw).F(jw)');
axis([-20 20 0 4]);
subplot(325);
plot(W,Y);
xlabel('w');
ylabel('Y(jw)');
```

```
axis([-20 20 0 4]);
```

$f(t),y(t),F(j\omega)$，$F(j\omega)\times F(j\omega)$及 $Y(j\omega)$ 的图形如图 9-11 所示。

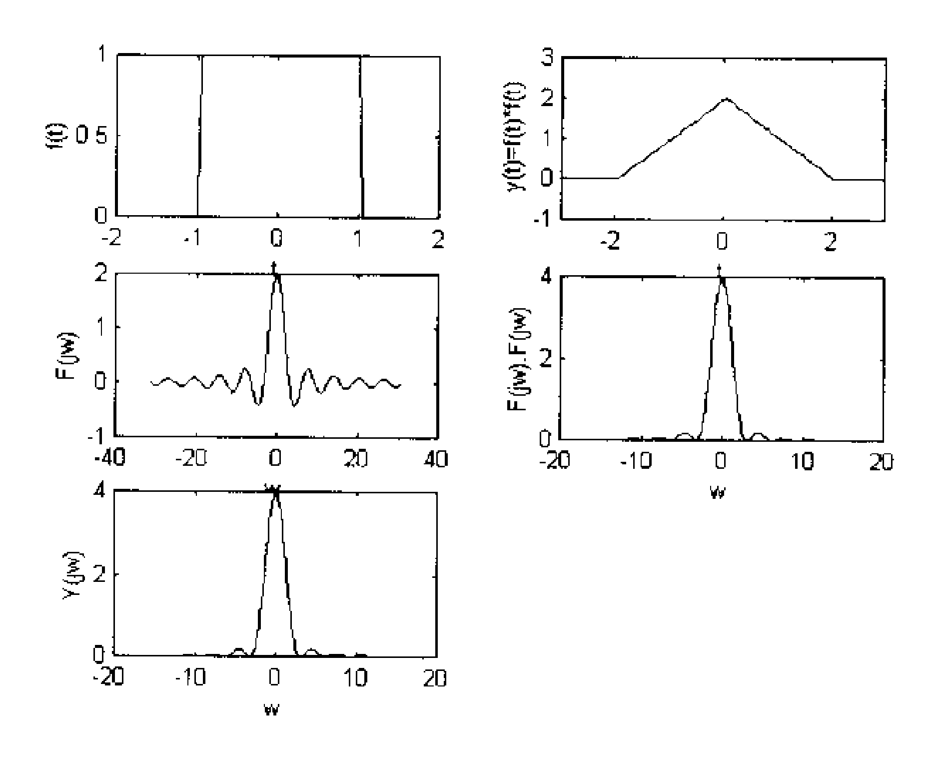


图 9-11　时域卷积定理示例

由图9-11可见，$Y(j\omega)$ 与 $F(j\omega)\times F(j\omega)$的图形一致，而$y(t)$的波形正是我们熟知的$f(t)*f(t)$的波形，$Y(j\omega)$ 也是熟知的$y(t)$的傅氏变换，从而验证时域卷积定理。

9.3.5　傅里叶变换的对称性

傅里叶变换的对称性为：

若 $f(t)\leftrightarrow F(j\omega)$，则：

$$F(jt)\leftrightarrow 2\pi f(-\omega) \tag{9-14}$$

下面举例说明付里叶变换的对称性。

例 9-13：设 $f(t)=Sa(t)$，已知信号 $f(t)$ 的傅里叶变换为 $F(j\omega)=\pi g_2(\omega)=\pi[\varepsilon(\omega+1)-\varepsilon(\omega-1)]$，用 MATLAB 求 $f_1(t)=\pi g_2(t)$ 的傅里叶变换 $F_1(j\omega)$，并验证对称性。

解：

MATLAB 程序为：

```
r=0.01;
t=-15:r:15;
f=sin(t)./t;
f1=pi*(Heaviside(t+1)-Heaviside(t-1));
```



```
N=500;
W=5*pi*1;
k=-N:N;
w=k*W/N;
F=r*sinc(t/pi)*exp(-j*t'*w);
F1=r*f1*exp(-j*t'*w);
subplot(221);plot(t,f);
xlabel('t');ylabel('f(t)');
subplot(222);plot(w,F);
axis([-2 2 -1 4]);
xlabel('w');ylabel('F(w)');
subplot(223);plot(t,f1);
axis([-2 2 -1 4]);
xlabel('t');ylabel('f1(t)');
subplot(224);plot(w,F1);
axis([-20 20 -3 7]);
xlabel('w');
ylabel('F1(w)');
```

程序运行结果如图 9-12 所示。

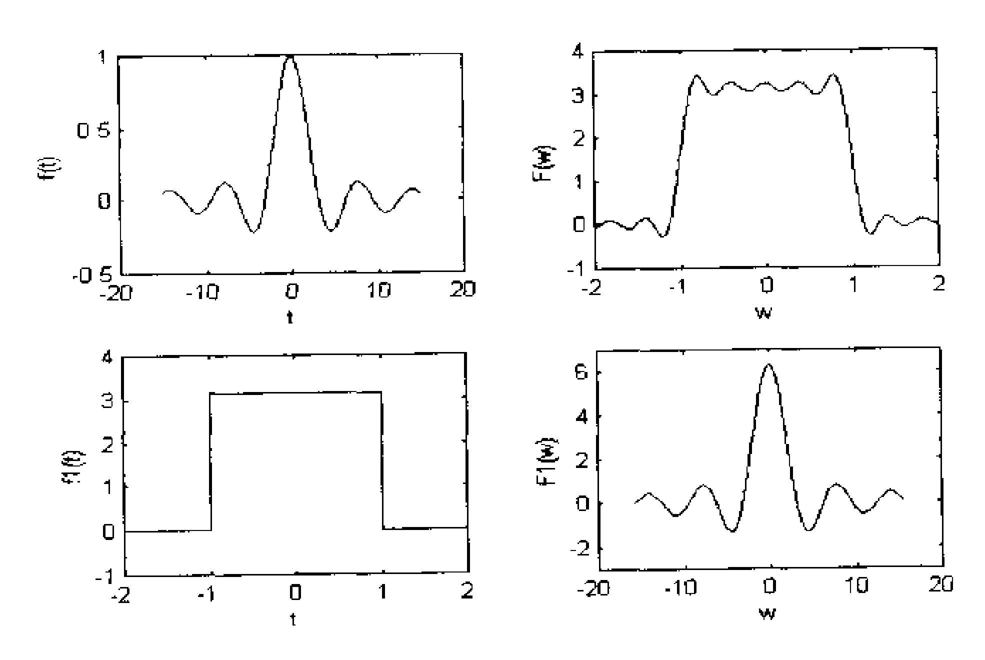


图 9-12 傅里叶变换对称性实例

由图 9-12 可见，$f(t)=Sa(t)$ 的傅里叶变换为 $F(j\omega)=\pi g_2(\omega)$，$F(jt)=f_1(t)=\pi g_2(t)$ 的傅里叶变换为 $F_1(j\omega)=2\pi Sa(\omega)=2\pi f(\omega)$，考虑到 $Sa(\omega)$ 是 ω 的偶函数，因此我们有：$F(jt)\leftrightarrow 2\pi f(-\omega)$，即验证了傅里叶变换的对称性。

由图 9-12 我们还可以看出，$F(j\omega)$ 的图形与门信号 $\pi g_2(t)$ 的波形有一定的误差，

这是因为在进行理论分析计算中，求本例的 $F(j\omega)$ 的过程相当于是采用原门信号 $\pi\ g_2(t)$ 的傅里叶变换（等于 $2\pi Sa(\omega)$）来恢复原信号，由于原信号 $\pi\ g_2(t)$ 在 $t=\pm1$ 处存在间断点，其恢复信号将产生过冲，这种现象即为著名的吉布斯现象。进一步的说明请参阅《信号与系统》的有关章节内容。

9.4.6 傅里叶变换的时域微分特性

傅里叶变换的时域微分特性为：

若 $f(t)\leftrightarrow F(j\omega)$，则：

$$\frac{\mathrm{d}f^n(t)}{\mathrm{d}t^n}\leftrightarrow(j\omega)^n F(j\omega) \tag{9-15}$$

下面举例说明傅里叶变换的一阶微分特性。

例 9-14：已知 $f(t)$ 的波形如图 9-13 所示，试用 MATLAB 求 $f(t)$ 及 $\frac{\mathrm{d}f(t)}{\mathrm{d}t}$ 的傅里叶变换，$F(j\omega)$ 及 $F_1(j\omega)$，并验证时域微分特性。

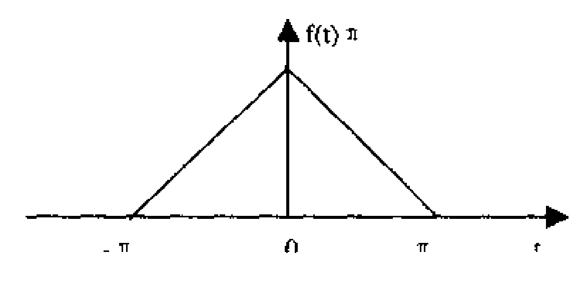


图 9-13 $f(t)$的波形

解：

在 MATLAB 中，有专门的三角波形生成函数 sawtooth()，其格式为：

```
f = sawtooth(t, width)
```

其中 width（0<width≤1 的标量）用于确定最大值的位置，即当 t 从 0～2π×width 变化时，f 从-1 上升到+1，然后当 t 从 2π×width 至 2π 时 $f(t)$又线性地从+1 下降到-1，周而复始。当 width=0.5 时，可产生一对称的标准三角波。利用此三角波与一门信号 $g_{2\pi}(t)$相乘，再进行必要的幅度调整（乘系数 π/2），并时移（左移 π）可得到 $f(t)$：

$$f(t)=\frac{\pi}{2}\cdot sawtooth(t+\pi,0.5)\times(Heaviside(t+\pi)-Heaviside(t-\pi))$$

又设 $f_1(t)=\frac{\mathrm{d}f(t)}{\mathrm{d}t}$，其波形如图 9-14 所示。

$f_1(t)$ 可用阶跃函数 *Heaviside*()生成：

$$f_1(t)=Heaviside(t+\pi)-2\cdot Heaviside(t)+Heaviside(t-\pi)$$

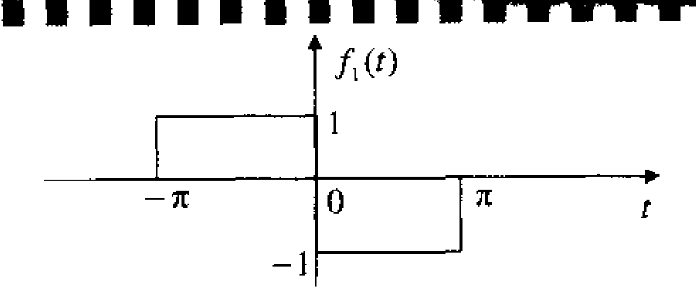


图 9-14 波形

在下列程序中，可直接求出 $F(j\omega)$ 及 $F_1(j\omega)$，因为 $F_1(j\omega)$ 是复函数，可分别求其幅度 abs(F_1)及相位 angle(F_1)。为验证时域微分特性，因为 $F_1(j\omega)=(j\omega)F(j\omega)$，也即 $F(j\omega)=\dfrac{F_1(j\omega)}{j\omega}$。在程序中将所求出的 $F_1(j\omega)$ 点除 $j\omega$，将其所得到的波形应与 $F(j\omega)$ 的波形一致。实现以上过程的程序如以下命令文件 sywf.m 所示：

```
r=0.01;
t=-5:r:5;
f1=Heaviside(t+pi)-Heaviside(t-pi);
f2=Heaviside(t+pi)-2*Heaviside(t)+Heaviside(t-pi);
f=pi/2*(sawtooth(t+pi,0.5)+1).*f1;
w1=2*pi*5;
N=200;
k=-N:N;
w=k*w1/N;
F=r*f*exp(-j*t'*w);
F2=r*f2*exp(-j*t'*w)
F3=F2./(j*w);
subplot(411);
plot(t,f2);
set(gca,'box','off')
xlabel('t');
ylabel('f2(t)');
subplot(412);
plot(t,f);
set(gca,'box','off')
xlabel('t');
ylabel('f(t)');
subplot(413);
plot(w,F);
set(gca,'box','off')
```

```
xlabel('w');
ylabel('F(jw)');
subplot(414);
plot(w,F3);
set(gca,'box','off')
xlabel('w');
ylabel('F3(jw)');
```

程序运行结果如图 9-15 所示。结果表明，$F(j\omega)$ 与 $\dfrac{F_1(j\omega)}{j\omega}=F_3(j\omega)$ 的曲线完全一致，从而验证了时域微分特性。

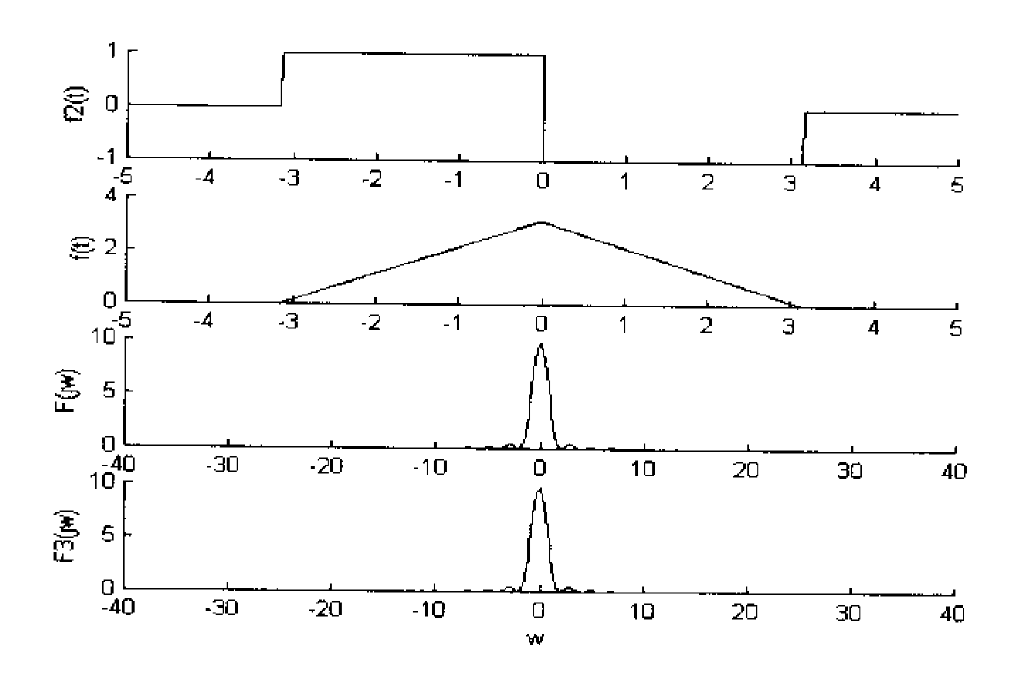


图 9-15　时域微分特性的例子

上机练习题

1．试用 fourier()函数求下列信号的傅里叶变换 $F(j\omega)$，并画出 $|F(j\omega)|$。

（1）　$f(t)=te^{-2t}\varepsilon(t)$；

（2）$f(t)=\mathrm{sgn}(t)=\begin{cases}1, & t>0\\ -1, & t<0\end{cases}$

2．试用 ifourier()函数求 $F(j\omega)=-j\dfrac{2\omega}{4^2+\omega^2}$ 的逆傅里叶变换并画出波形。



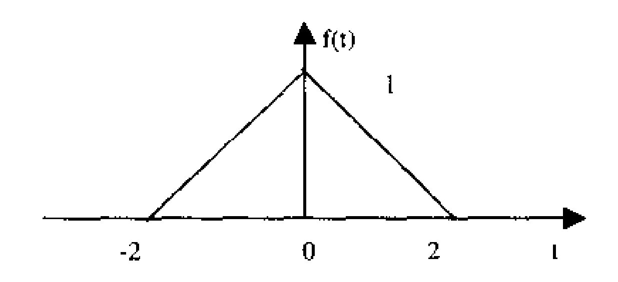
3．设 $f(t)=\begin{cases}1-\dfrac{|t|}{2} & |t|<2\\ 0 & |t|>2\end{cases}$，要求：

（1）求 $f(t)$ 的符号表达式；

（2）用 ezplot()绘出 $f(t)$；

（3）当用 fourier()函数求 $f(t)$ 的傅里叶变换 $F(j\omega)$ 时，出现什么情况？讨论能否用 ezplot()绘出 $|F(j\omega)|$。

4．$f_1(t)$ 的波形如下图所示。



又 $f(t)=f_1(t-2)\cos(100t)$，试用连续信号的傅里叶变换数值算法，求：

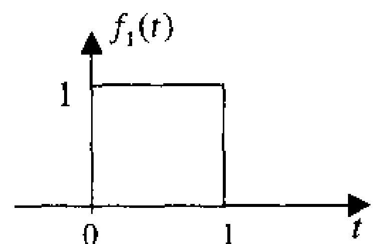
（1）$f_1(t)$ 的傅里叶变换 $F_1(j\omega)$ 的 $|F_1(j\omega)|$ 及 $\varphi_1(\omega)$；

（2）$f_1(t-2)$ 的傅里叶变换 $F_2(j\omega)$ 的 $|F_2(j\omega)|$ 及 $\varphi_2(\omega)$；

（3）$f(t)$ 的傅里叶变换 $F(j\omega)$ 的 $|F(j\omega)|$ 及 $\varphi(\omega)$；

分别比较 $|F_1(j\omega)|$、$|F_2(j\omega)|$、$|F(j\omega)|$ 及 $\varphi_1(\omega)$、$\varphi_2(\omega)$、$\varphi(\omega)$，验证傅里叶变换的相关特性。

5．已知 $f_1(t)$ 的波形如下图所示。



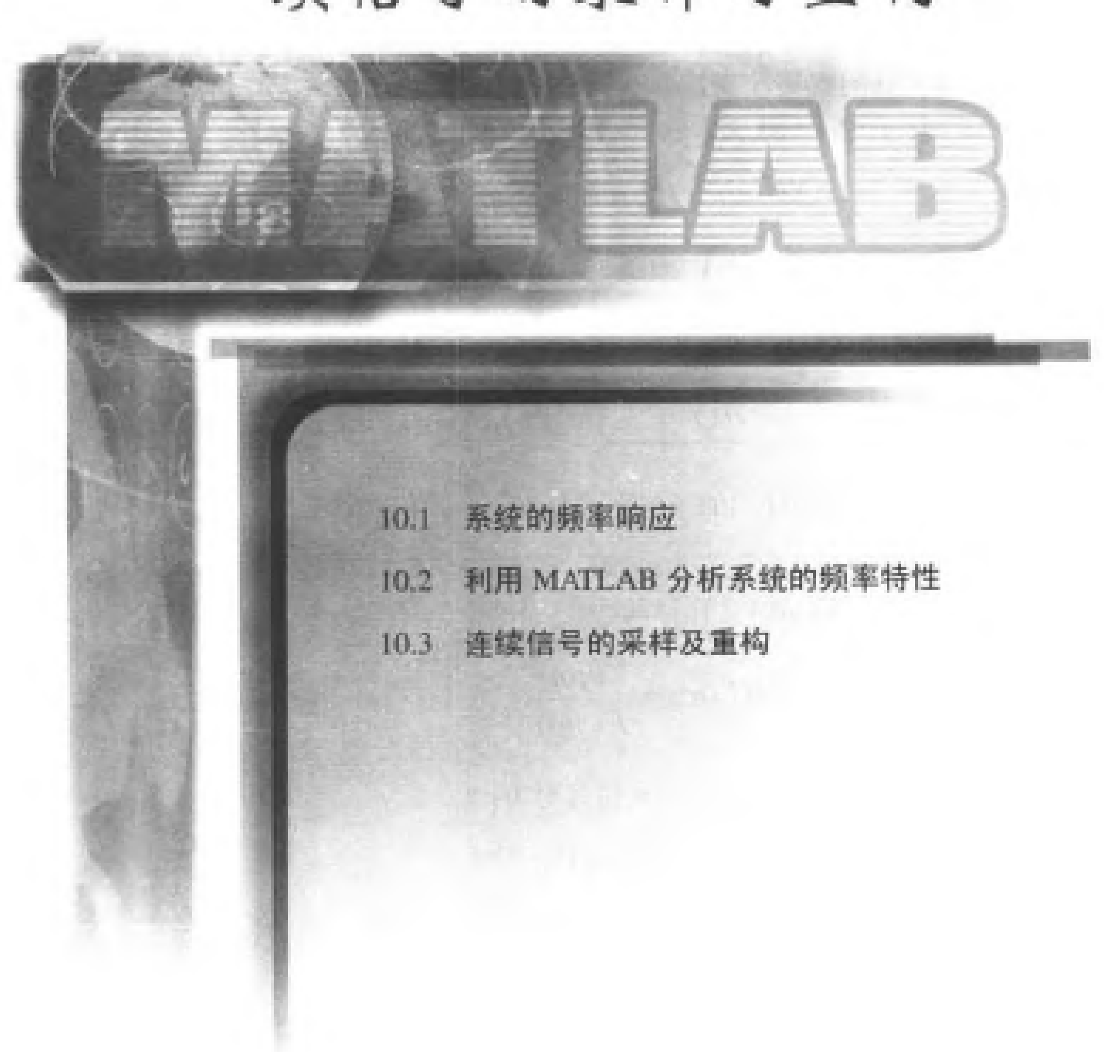
且 $f_1(t)\leftrightarrow F_1(j\omega)$；设 $f(t)=f_1(t)*f_1(t)\leftrightarrow F(j\omega)$，试用 MATLAB 给出 $f_1(t)$、$f(t)$、$F_1(j\omega)$ 及 $F(j\omega)$，并验证时域卷积定理。



日期：__________ 进程：__________ 备注：__________

http://www.facit.com.cn E-mail:[illegible]
Tel: (010) 68134545 6813[illegible]

第 10 章　连续系统的频域分析及连续信号的采样与重构





10.1　系统的频率响应

设线性时不变（LTI）系统的冲激响应为$h(t)$，该系统的输入（激励）信号为$f(t)$，则此系统的零状态输出（响应）$y(t)$为：

$$y(t)=h(t)*f(t) \tag{10-1}$$

式（10-1）是$f(t),h(t)$及$y(t)$在时域上的关系式。

又设$f(t),h(t)$及$y(t)$的傅里叶变换分别为$F(j\omega),H(j\omega)$及$Y(j\omega)$，根据时域卷积定理，与（10-1）式对应的$F(j\omega),H(j\omega)$及$Y(j\omega)$在频域上的关系式为：

$$Y(j\omega)=H(j\omega)\cdot F(j\omega) \tag{10-2}$$

（10-1）式及（10-2）式的原理框图如图 10-1 所示。

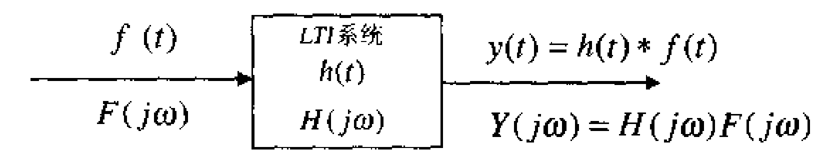


图 10-1　LTI 系统的时域及频域分析图

一般地，连续系统的频率响应定义为系统的零状态响应$y(t)$的傅里叶变换$Y(j\omega)$与输入信号$f(t)$的傅里叶变换$F(j\omega)$之比，即：

$$H(j\omega)=\frac{Y(j\omega)}{F(j\omega)} \tag{10-3}$$

通常，$H(j\omega)$是ω的复函数，因此，又将其写为：

$$H(j\omega)=|H(j\omega)|e^{j\varphi(\omega)} \tag{10-4}$$

如果令$Y(j\omega)=|Y(j\omega)|e^{j\varphi_y(\omega)}$，$F(j\omega)=|F(j\omega)|e^{j\varphi_f(\omega)}$，则有：

$$|H(j\omega)|=\frac{|Y(j\omega)|}{|F(j\omega)|} \tag{10-5}$$

$$\varphi(\omega)=\varphi_y(\omega)-\varphi_f(\omega) \tag{10-6}$$

称$|H(j\omega)|$为系统的幅频响应，$\varphi(\omega)$为系统的相频响应。

需要提醒的是，$H(j\omega)$是系统的固有属性，它与激励信号$f(t)$的具体形式无关。求系统的$H(j\omega)$，当然可以按照（10-3）式的定义求，但在工程实际中往往是给出具体的系统图（如具体的电路形式），通过电路分析的方法直接求出$H(j\omega)$。

通常，$H(j\omega)$可表示成两个有理多项式$B(j\omega)$与$A(j\omega)$的商，即：



$$H(j\omega)=\frac{B(j\omega)}{A(j\omega)}=\frac{b_1(j\omega)^m+b_2(j\omega)^{m-1}+\cdots+b_{m-1}(j\omega)+b_m}{a_1(j\omega)^n+a_2(j\omega)^{n-1}+\cdots+a_{n-1}(j\omega)+a_n} \tag{10-7}$$

10.2　利用 MATLAB 分析系统的频率特性

MATLAB 提供了专门对连续系统频率响应 $H(j\omega)$ 进行分析的函数 freqs()。该函数可以求出系统频率响应的数值解，并可绘出系统的幅频及相频响应曲线。freqs()函数有如下四种调用格式：

1．h=freqs(b,a,w)

该调用格式中，b 为对应于（10-7）式的向量$[b_1,b_2,b_3,\cdots b_m]$，a 为对应于（10-7）式的向量$[a_1,a_2,a_3,\cdots a_n]$，w 为形如 w1:p:w2 的冒号运算定义的系统频率响应的频率范围，$w1$ 为频率起始值，$w2$ 为频率终止值，p 为频率取样间隔。向量 h 则返回在向量 w 所定义的频率点上，系统频率响应的样值。

例如，运行如下命令：

```
a=[1 2 1];
b=[0 1];
h=freqs(b,a,0:0.5:2*pi)          %计算 0~2π 频率范围内以间隔 0.5 取样的系统频率响应的样值
```

则程序运行结果为：

```
h =
  Columns 1 through 4
   1.0000              0.4800 - 0.6400i           0 - 0.5000i    -0.1183 - 0.2840i
  Columns 5 through 8
  -0.1200 - 0.1600i   -0.0999 - 0.0951i   -0.0800 - 0.0600i   -0.0641 - 0.0399i
  Columns 9 through 12
  -0.0519 - 0.0277i   -0.0426 - 0.0199i   -0.0355 - 0.0148i   -0.0300 - 0.0113i
  Column 13
  -0.0256 - 0.0088i
```

2．[h,w]=freqs(b,a)

该调用格式将计算默认频率范围内 200 个频率点的系统频率响应的样值，并赋值给返回变量 h，200 个频率点记录在 w 中。

3．[h,w]=freqs(b,a ,n)

该调用格式将计算默认频率范围内 n 个频率点上系统频率响应的样值，并赋值给返回变量 h，n 个频率点记录在 w 中。



4．freqs(b,a)

该格式并不返回系统频率响应的样值，而是以对数坐标的方式绘出系统的幅频响应和相频响应曲线。例如运行如下命令：

```
a=[1 0.4 1];
b=[1 0 0];
freqs(b,a)
```

程序运行结果如图 10-2 所示。

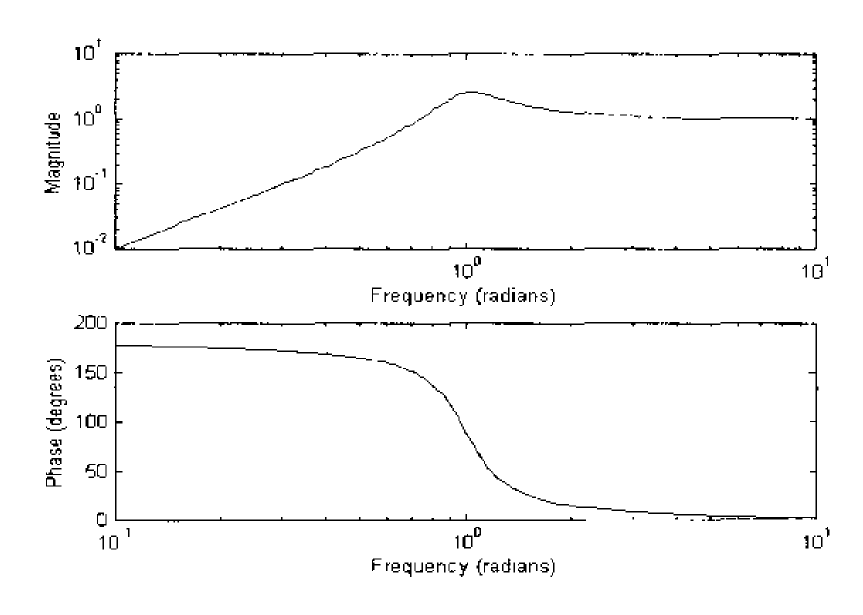


图 10-2 对数坐标的系统幅频响应及相频响应曲线

下面通过具体例子说明调用 freqs()函数求系统频率响应。

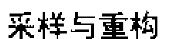
例 10-1：理想低通滤波器在物理上是不可实现的，但传输特性近似于理想特性的电路却能找到。图 10-3 是常见的用 RLC 元件构成的二阶低通滤波器（一般说来，阶数越高，实际滤波器的特性越接近于理想特性），该电路的频率响应 $H(j\omega)$ 为：

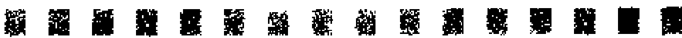
$$H(j\omega)=\frac{U_R(j\omega)}{U_S(j\omega)}=\frac{1}{1-\omega^2LC+j\omega\dfrac{L}{R}}$$

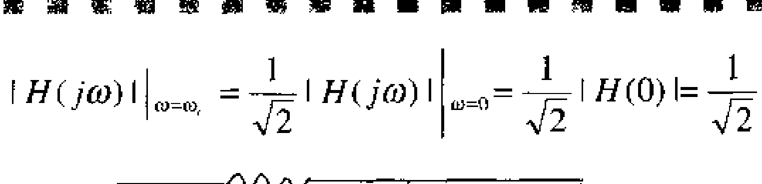
设 $R=\sqrt{\dfrac{L}{2C}}$，$L=0.8H, C=0.1F, R=2\Omega$，试用 MATLAB 的 freqs 函数绘出该频率响应，如图 10-3 所示。

解：

令截止频率 $\omega_C=\dfrac{1}{\sqrt{LC}}$，当 $\omega=\omega_c=\dfrac{1}{\sqrt{0.08}}=3.54$ 时。







$$|H(j\omega)|\Big|_{\omega=\omega_c}=\frac{1}{\sqrt{2}}|H(j\omega)|\Big|_{\omega=0}=\frac{1}{\sqrt{2}}|H(0)|=\frac{1}{\sqrt{2}}$$

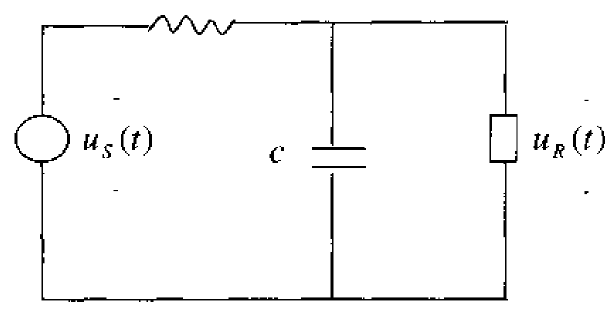


图 10-3　RLC 二阶低通滤波器电路路图

将 L，C，R 的值代入 $H(j\omega)$ 的表达式，得：

$$H(j\omega)=\frac{1}{0.08(j\omega)^2+0.4j\omega+1}=\left|H(j\omega)e^{j\varphi(\omega)}\right|$$

其中：

$$|H(j\omega)|=\frac{1}{\sqrt{1+(\frac{\omega}{\omega_C})^4}}=\frac{1}{\sqrt{1+0.08^2\omega^4}}$$

$$\varphi(\omega)=-\arctan\left[\frac{\sqrt{2}(\frac{\omega}{\omega_C})}{1-(\frac{\omega}{\omega_C})^2}\right]=-\arctan\left[\frac{\sqrt{2\times0.08}\omega}{1-0.08\omega^2}\right]$$

实现该系统响应的程序为下列命令文件：

```
b=[0 0 1];                    %生成向量 b
a=[0.08 0.4 1];               %生成向量 a
[h,w]=freqs(b,a,100);         %求系统响应函数 H(jω)，设定 100 个频率点
h1=abs(h);                    %求幅频响应
h2=angle(h);                  %求相频响应
subplot(211);
plot(w,h1);
grid
xlabel('角频率(W)');
ylabel('幅度');
title('H(jw)的幅频特性');
subplot(212);
```

```
plot(w,h2*180/pi);
grid
xlabel('角频率(w)');
ylabel('相位(度)');
title('H(jw)的相频特性');
```

程序运行结果如图 10-4 所示。

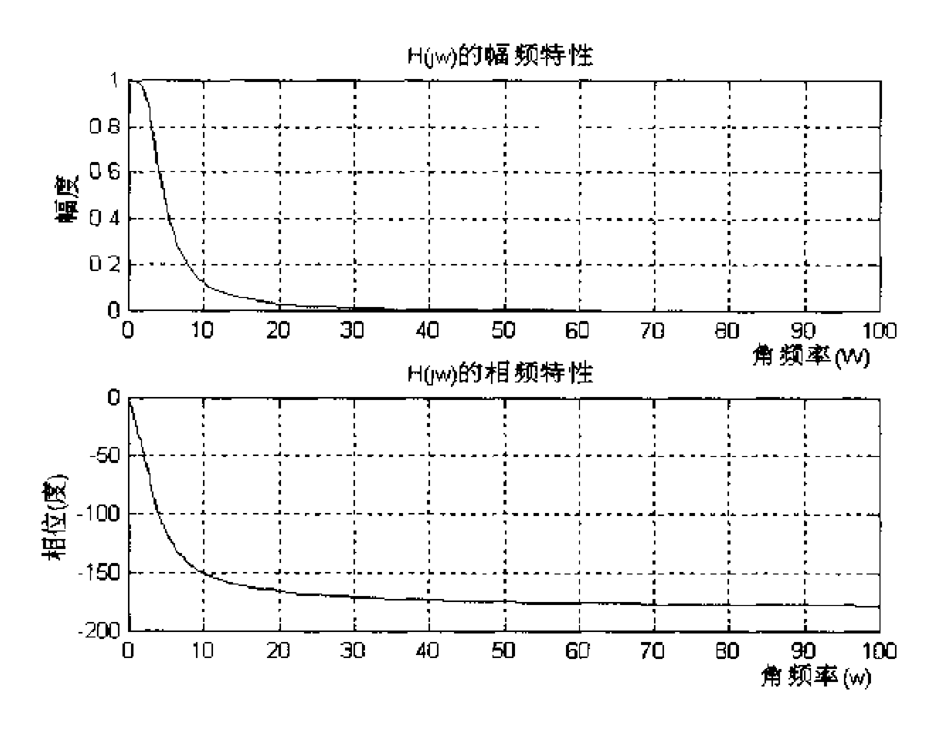


图 10-4　RLC 二阶低通滤波器的幅频特性及相频特性

由图 10-4 可见，当 ω 从 0 增大时，该低通滤波器的幅度从 1 降到 0，ω_c 约为 3.5；而 $\varphi(\omega)$ 从 0^0 降到-180^0，与我们理论分析的结果一致。

例 10-2：全通网络是指其系统函数 $H(j\omega)$ 的极点位于左半平面，零点位于右半平面，且零点与极点对于 $j\omega$ 轴互为镜像对称的网络。它可保证不影响传输信号的幅频特性，只改变信号的相频特性。图 10-5 是用 RLC 构成的格形滤波器，当满足 $\frac{L}{C}=R^2$ 时即构成全通网络。其 $H(j\omega)=\frac{U_2(j\omega)}{U_1(j\omega)}=\frac{R-j\omega L}{R+j\omega L}$，设 $R=10\Omega$，$L=2H$，试用 MATLAB 求 $|H(j\omega)|$ 及 $\varphi(\omega)$。

解：

将 R、L 的值代入 $H(j\omega)$ 中，得：

$$H(j\omega)=\frac{-2j\omega+10}{2j\omega+10}$$

$$|H(j\omega)|=1$$



$$\varphi(\omega)=-2\arctan(\frac{\omega L}{R})=-2\arctan(\frac{\omega}{5})$$

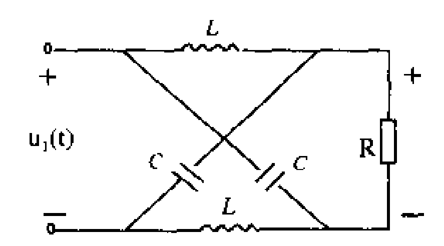


图 10-5 格形滤波器

将例 10-1 的程序中的 a、b 向量改为：

```
b=[-2 10];a=[2 10];
```

即可求出本例的$|H(j\omega)|$及$\varphi(\omega)$。

程序运行结果如图 10-6 所示。

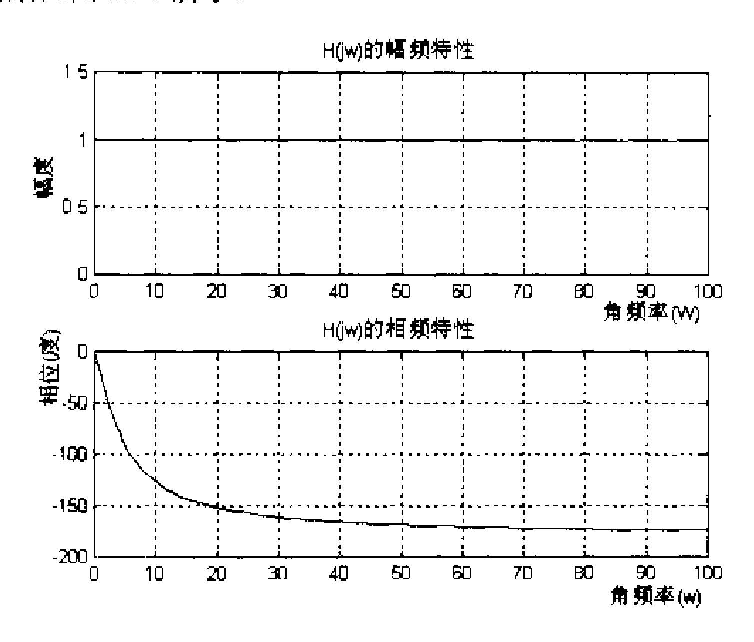


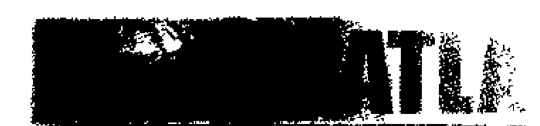
图 10-6 全通网络的幅频特性及相频特性

由图 10-6 可见，当ω从 0 增大时，$H(j\omega)$的幅频特性是一条数值为 1 的水平线，即对输入信号的各频率分量都进行等值传输；而$\varphi(\omega)$从 0^0 开始下降，最终趋于-180^0。这种网络称为全通网络，在传输系统中常用来进行相位校正，如作为相位均衡器或移相器。

10.3 连续信号的采样及重构

10.3.1 信号的采样

图 10-7 给出了信号采样的原理图。



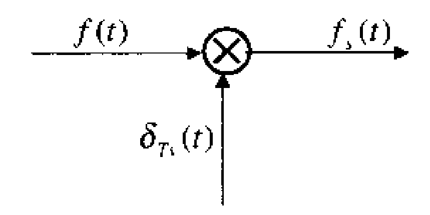


图 10-7 信号采样原理图

由图 10-7 可见，$f_s(t)=f(t)\cdot\delta_{T_s}(t)$，其中，冲激采样信号 $\delta_{T_s}(t)$ 的表达式为：

$$\delta_{Ts}(t)=\sum_{n=-\infty}^{\infty}\delta(t-nT_s) \tag{10-8}$$

其傅里叶变换为 $\omega_S\sum_{n=-\infty}^{\infty}\delta(\omega-n\omega_S)$，其中 $\omega_S=\dfrac{2\pi}{T_s}$，设 $F(j\omega)$ 为 $f(t)$ 的傅里叶变换，图 10-8 给出了采样过程的时域波形变化图及其频谱。

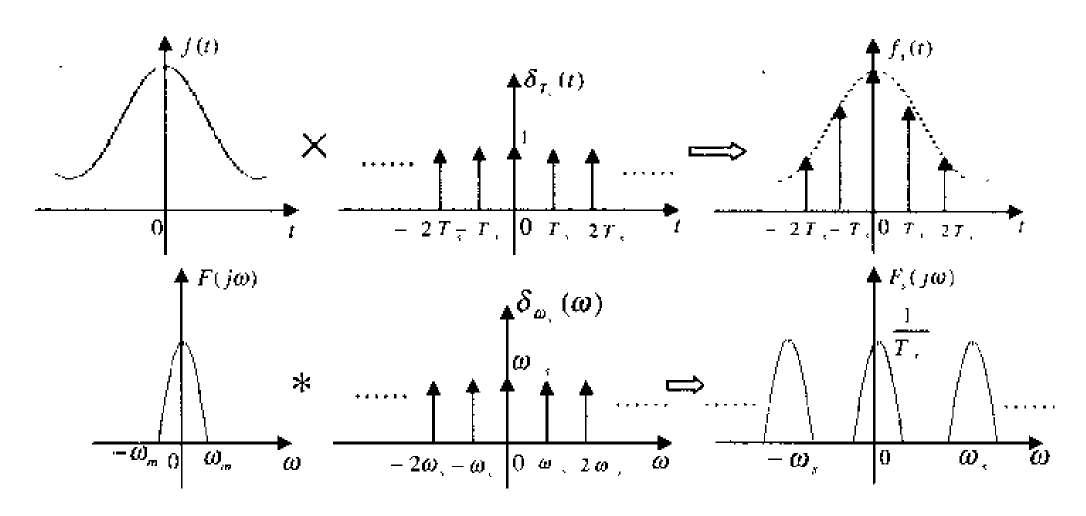


图 10-8 冲激采样及其频谱

设 $f_s(t)$ 的频谱为 $F_S(j\omega)$，由傅里叶变换的频域卷积定理，有：

$$f_s(t)=f(t)\delta_{T_s}(t)\leftrightarrow F_S(j\omega)=\frac{1}{2\pi}F(j\omega)*\omega_S\sum_{n=-\infty}^{\infty}\delta(\omega-n\omega_S)=\frac{1}{T_S}\sum_{n=-\infty}^{\infty}F[j(\omega-\omega_S)] \tag{10-9}$$

若设 $f(t)$ 是带限信号，带宽为 ω_m，即当 $|\omega|>\omega_m$ 时，$f(t)$ 的频谱 $F(j\omega)$ 的值为 0，则由式（10-9）可见，$f(t)$ 经采样后的频谱 $F_S(j\omega)$ 就是将 $F(j\omega)$ 在频率轴上搬移至 $0,\pm\omega_S,\pm2\omega_S,\cdots,\pm n\omega_S,\cdots$ 处（幅度为原频谱的 $1/Ts$ 倍）。因此，当 $\omega_S\geqslant2\omega_m$ 时，频谱不发生混叠；而当 $\omega_S<2\omega_m$ 时，频谱发生混叠。图 10-9 给出了以上这两种情况的波形图。

我们选取信号 $f(t)=Sa(t)$ 作为被采样的信号，这是因为：第一，$f(t)$ 是一个带限



信号，其$\omega_m=1$；第二，它是一个典型的信号，是分析其他信号的基础，因此完全有必要对它的信号特征详加了解。此外，应该指出的是，实际信号中，绝大多数都不是严格意义上的带限信号，这时根据实际精度要求来确定信号的带宽ω_m。

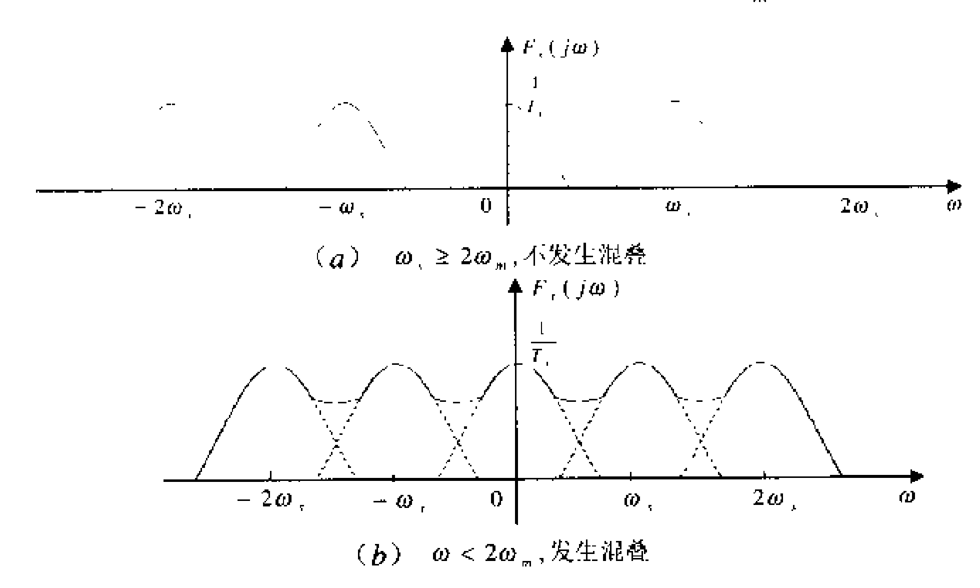


图 10-9　采样信号的频谱

例 10-3：当采样频率$\omega_S=2\omega_m$时，称为临界采样，取$\omega_c=\omega_m$。下列程序实现对信号$Sa(t)$的采样及由该采样信号恢复$Sa(t)$（见 10.3.2 小节）。

```
wm=1;                    %信号带宽
wc=wm;                   %滤波器截止频率
Ts=pi/wm;                %采样间隔
ws=2*pi/Ts;              %采样角频率
n=-100:100;              %时域采样点数
nTs=n*Ts                 %时域采样点
f=sinc(nTs/pi);
Dt=0.005;t=-15:Dt:15;
fa=f*Ts*wc/pi*sinc((wc/pi)*(ones(length(nTs),1)*t-nTs'*ones(1,length(t))));    %信号重构
t1=-15:0.5:15;
f1=sinc(t1/pi);
subplot(211);
stem(t1,f1);
xlabel('kTs');
ylabel('f(kTs)');
title('sa(t)=sinc(t/pi)的临界采样信号');
subplot(212);
plot(t,fa)
xlabel('t');
```

```
ylabel('fa(t)');
title('由 sa(t)=sinc(t/pi)的临界采样信号重构 sa(t)');
grid;
```

程序运行结果如图 10-10 所示。

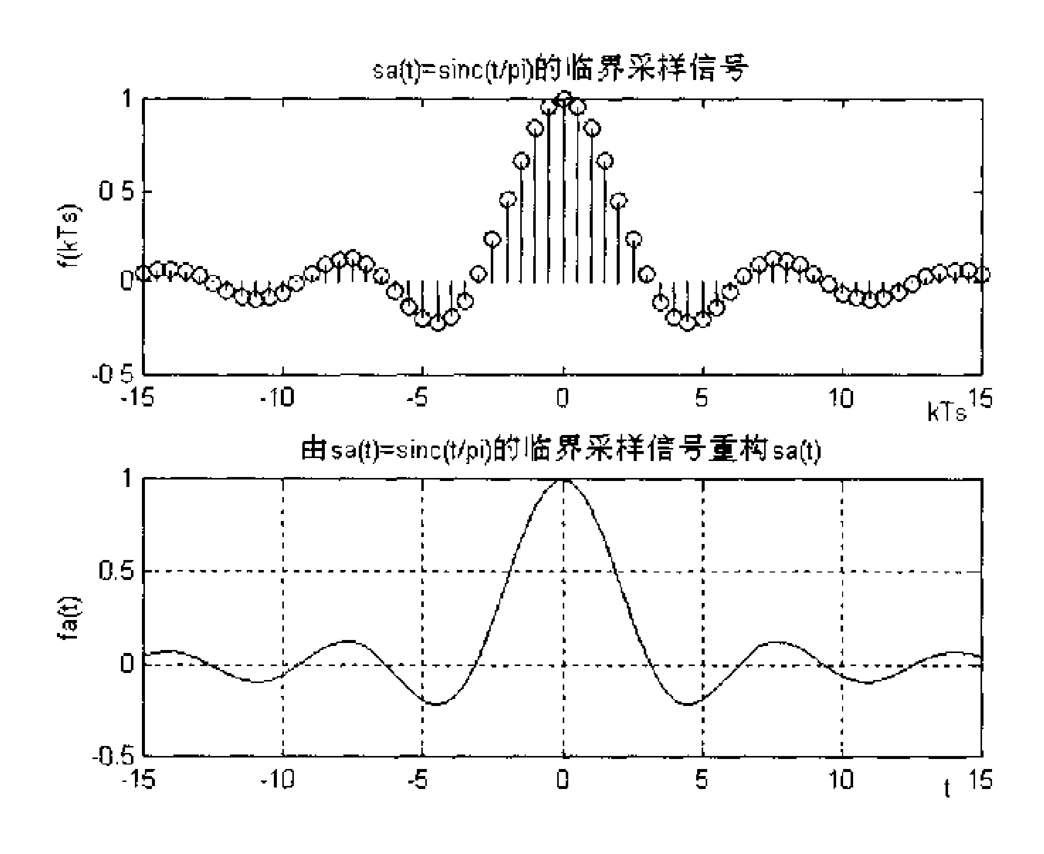


图 10-10　临界采样信号及信号恢复

10.3.2　信号的重构

设信号 $f(t)$ 被采样后所形成的采样信号为 $f_S(t)$，信号的重构是指由 $f_S(t)$ 经内插处理后，恢复出原来的信号 $f(t)$ 的过程。因此又称为信号恢复。

设 $f(t)$ 为带限信号，带宽为 ω_m，经采样后的频谱为 $F_s(j\omega)$。设采样频率 $\omega_S \geqslant 2\omega_m$，则由式（10-9）知 $F_S(j\omega)$ 是以 ω_S 为周期的谱线（见图 10-11（a））。现选取一个频率特性 $H(j\omega)=\begin{cases}T_s, & |\omega|<\omega_c\\ 0, & |\omega|>\omega_c\end{cases}$（见图 10-11（$b$），其中，截止频率 ω_C 满足 $\omega_m \leqslant \omega_C \leqslant \dfrac{\omega_S}{2}$）的理想低通滤波器与 $F_S(j\omega)$ 相乘，得到的频谱即为原信号的频谱 $F(j\omega)$（见图 10-11（c）），对应的时域波形分别为图 10-11（d）、（e）、（f）。实现的原理图如图 10-12 所示。



显然，$F(j\omega)=H(j\omega)\cdot F_S(j\omega)$，与之对应的时域表达式为 $f(t)=h(t)*f_S(t)$。

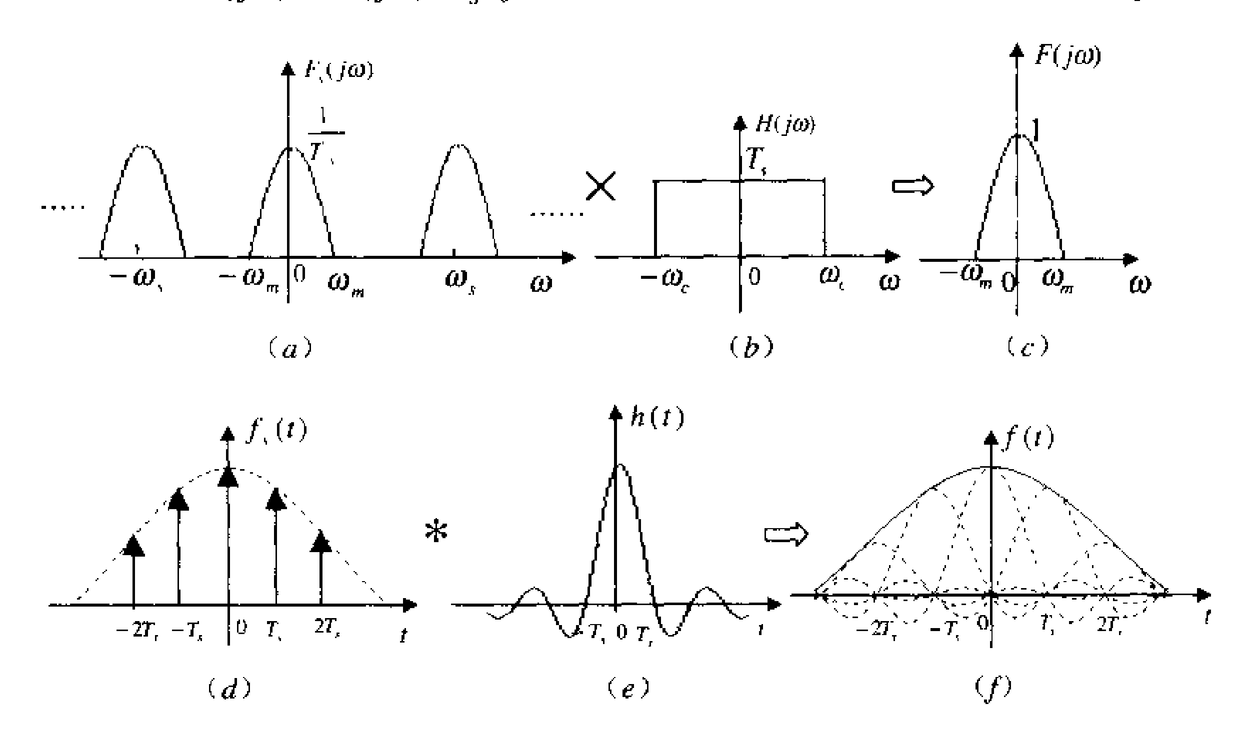


图 10-11　$f_s(t)$ 经理想低通滤波器重构的信号 $f(t)$

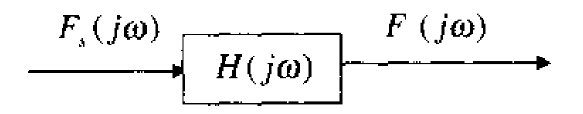


图 10-12　低通滤波器恢复原信号

通过以上分析，得到如下的时域取样定理：

一个带宽为 ω_m 的带限信号 $f(t)$，可惟一地由它的均匀取样信号 $f_S(t)=f(nT_S)$ 确定，其中，取样间隔 $T_S<\frac{\pi}{\omega_m}$，该取样间隔又称为奈奎斯特（Nyquist）间隔。

下面我们求出由 $f(nT_S)$ 构成 $f(t)$ 的表达式。

根据时域卷积定理，有：

$$f(t)=h(t)*f_S(t) \tag{10-10}$$

而

$$f_S(t)=f(t)\cdot\sum_{n=-\infty}^{\infty}\delta(t-nT_s)=\sum_{n=-\infty}^{\infty}f(nT_S)\delta(t-nT_S)$$



$$h(t) = F^{-1}[H(j\omega)] = T_S \frac{\omega_C}{\pi} Sa(\omega_C t)$$

其中 ω_C 为 $H(j\omega)$ 的截止角频率。将 $h(t)$ 及 $f_s(t)$ 代入式（10-10），得：

$$f(t) = f_s(t) * T_s \frac{\omega_c}{\pi} Sa(\omega_c t) = \frac{T_s\omega_c}{\pi} \sum_{n=-\infty}^{\infty} f(nT_s) Sa[\omega_c(t - nT_s)] \tag{10-11}$$

式（10-11）即为用 $f(nT_S)$ 表达 $f(t)$ 的表达式，是下面用 MATLAB 实现信号重构的基本关系式。顺便指出，抽样函数 $Sa(\omega_c t)$ 在此起着内插函数的作用，又称为内插函数。

例 10-4：设 $f(t) = Sa(t) = \dfrac{\sin t}{t}$，其 $F\ (j\omega)$ 为：

$$H(j\omega) = \begin{cases} \pi, & |\omega| < 1 \\ 0, & |\omega| > 1 \end{cases}$$

即 $f(t)$ 的带宽 $\omega_m = 1$，为了由 $f(t)$ 的采样信号 $f_S(t)$ 不失真地重构 $f(t)$，由时域采样定理知采样间隔 $T_S < \dfrac{\pi}{\omega_m} = \pi$，取 $T_S = 0.7\pi$（过采样）。利用 MATLAB 中的抽样函数 $Sinc(t) = \dfrac{\sin(\pi t)}{\pi t}$ 来表示 $Sa(t)$，有 $Sa(t) = Sinc(\dfrac{t}{\pi})$。据此由（10-11）式可知：

$$f(t) = T_s \frac{\omega_c}{\pi} \sum_{n=-\infty}^{\infty} f(nT_s) Sinc\left[\frac{\omega_c}{\pi}(t - nT_s)\right] \tag{10-12}$$

为了比较由采样信号恢复后的信号与原信号的误差，计算两信号的绝对误差。MATLAB 实现此过程的程序为如下所示的命令文件。

```
wm=1;                          %信号带宽
wc=1.1*wm;                     %滤波器截止频率
Ts=0.7*pi/wm;                  %采样间隔
ws=2*pi/Ts;                    %采样角频率
n=-100:100;
nTs=n*Ts
f=sinc(nTs/pi);
Dt=0.005;t=-15:Dt:15,
fa=f*Ts*wc/pi*sinc((wc/pi)*(ones(length(nTs),1)*t-nTs'*ones(1,length(t))));
%信号重构
error=abs(fa-sinc(t/pi)); %求重构信号与原信号的误差
t1=-15:0.5:15;
f1=sinc(t1/pi);
subplot(311);
```



```
stem(t1,f1);
xlabel('kTs');
ylabel('f(kTs)');
title('sa(t)=sinc(t/pi)的采样信号');
subplot(312);
plot(t,fa)
xlabel('t');
ylabel('fa(t)');
title('由 sa(t)=sinc(t/pi)的过采样信号重构 sa(t)');
grid;
subplot(313);
plot(t,error);
xlabel('t');
ylabel('error(t)');
title('过采样信号与原信号的误差 error(t)');
```

结果如图 10-13 所示。由图 10-13 可知，两信号的绝对误差 error 已在10^{-6}数量级，说明重构信号的精度已很好。

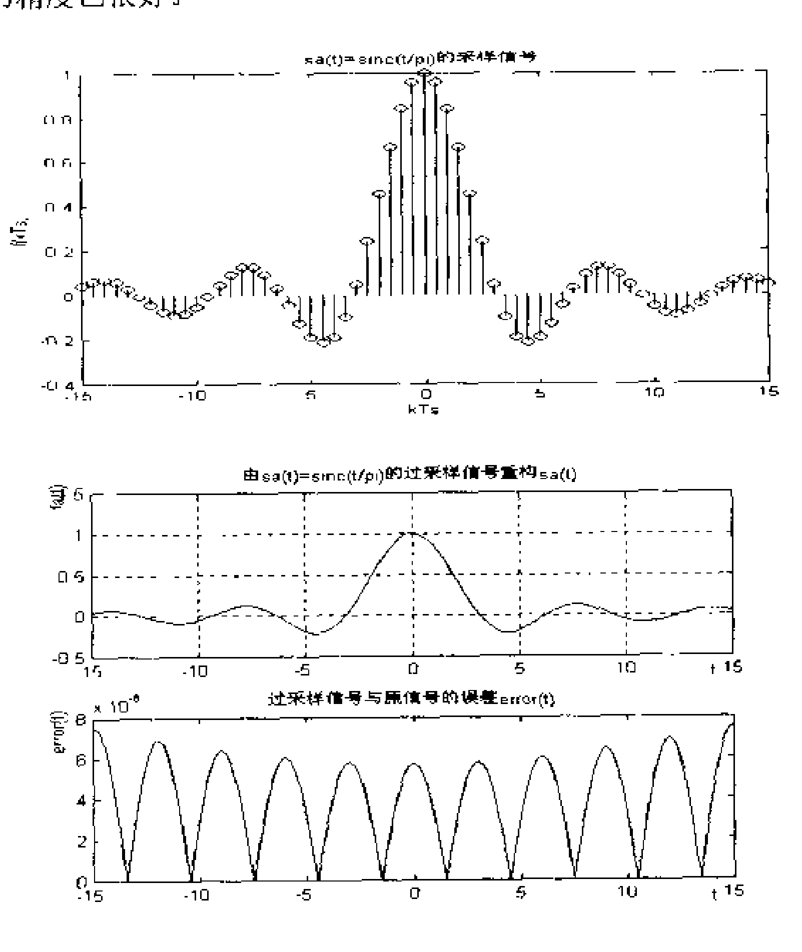


图 10-13　过采样信号、重构信号及两信号的绝对误差

例 10-5：设 $f(t)=Sa(t)=\dfrac{\sin t}{t}$，取样间隔 $T_s=1.5\pi$，对 $f(t)$ 进行采样（称为欠采样）。试由该采样信号重构原信号并求两信号的绝对误差。

解：

将例 10-4 的程序稍加改动，令 $\omega_m=1,\omega_c=\omega_m T_s=1.5\pi$，其余不变。改动后程序运行的结果如图 10-14 所示。

由图 10-14 可见，绝对误差 error 已大为增大，其原因是因采样信号的频谱混叠，使得在 $|\omega|<\omega_c$ 区域内的频谱相互"干扰"所致。

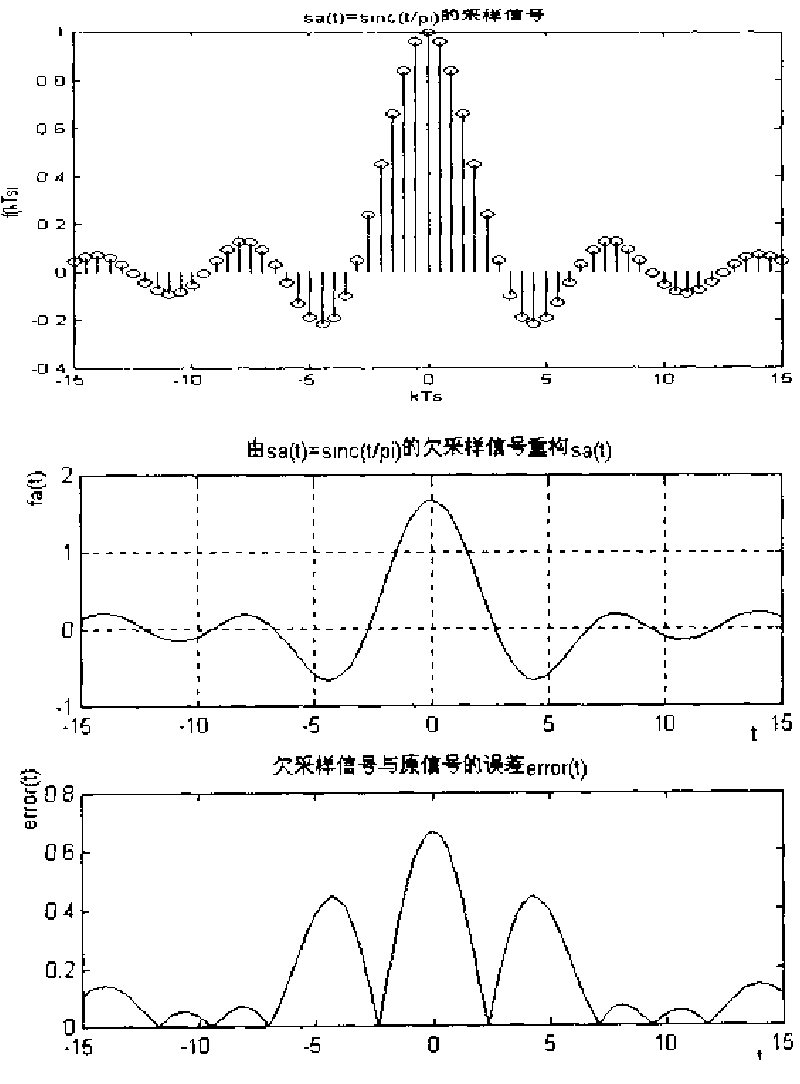


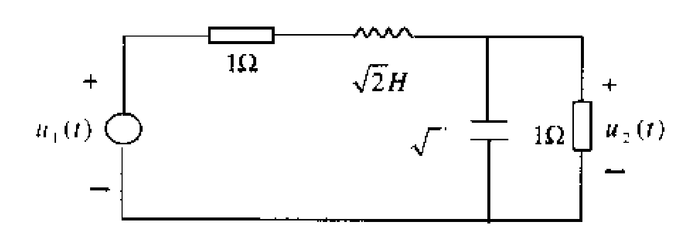
图 10-14 欠采样信号、重构信号及两信号的绝对误差

上机练习题

1．如下图所示的电路为最平幅度型二阶低通滤波器。试用 MATLAB 程序画出系统响

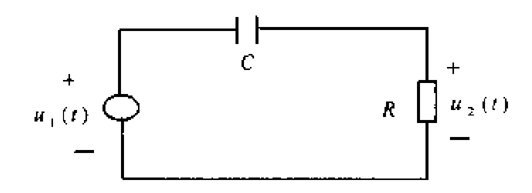


应 $H(j\omega)=\dfrac{U_2(j\omega)}{U_1(j\omega)}$ 的幅度响应及相频响应，并与理论分析的结果进行比较。$H(j\omega)$ 的截止频率 ω_0=?



2. 设 $H(j\omega)=\dfrac{2j\omega}{(j\omega+1)^2+100^2}$，试用 MATLAB 画出 $|H(j\omega)|$ 及 $\varphi(\omega)$，程序中建议使用 axis()确定观察窗口，使曲线更为清晰。该 $H(j\omega)$ 具有什么样的滤波特性？

3. 由 RC 构成的一阶高通滤波器如下图所示。设 $R=100\,\Omega$，$C=2F$，求 $H(j\omega)=\dfrac{U_2(j\omega)}{U_1(j\omega)}$，并用 MATLAB 画出 $|H(j\omega)|$ 及 $\varphi(\omega)$，并求 $H(j\omega)$ 的高通截止频率 ω_0=?，请与理论计算结果相比较。



4. 设 $f(t)=e^{-1000|t|}$，由于不是严格的带限信号，但其带宽 ω_m 可根据一定的精度要求做一近似。试根据以下三种情况用 MATLAB 实现由 $f(t)$ 的抽样信号 $f_s(t)$ 重构 $f(t)$ 并求两者误差，分析三种情况下的结果。

（1）$\omega_m=5000\pi,\omega_c=\omega_m,T_s=\pi/\omega_m$；

（2）$\omega_m=10000\pi,\omega_c=1.1\omega_m,T_s=\pi/\omega_m$；

（3）$\omega_m=2500\pi,\omega_c=0.9\omega_m,T_s=\pi/\omega_m$。



日期：________ 进度：________ 备注：________

第 11 章　连续系统的复频域分析及 MATLAB 实现



11.1 拉普拉斯变换及其曲面图

11.1.1 用 MATLAB 绘制拉普拉斯变换的曲面图

拉普拉斯变换是分析连续时间信号的有效手段，对于当$t \to \infty$时信号幅度不衰减或增长的时间信号，其傅里叶变换不存在，但我们可以用拉普拉斯变换来分析它们。

连续时间信号$f(t)$的拉普拉斯变换定义为：

$$F(s)=\int_{-\infty}^{\infty} f(t)e^{-st}\,\mathrm{d}t \qquad (11\text{-}1)$$

其中$s=\sigma+j\omega$，若以σ为横坐标（实轴），$j\omega$为纵坐标（虚轴），复变量s就构成了一个复平面，称为s平面。

显然，$F(s)$是复变量s的复函数，为了便于理解和分析$F(s)$随s的变化规律，我们可以将$F(s)$写成：

$$F(s)=|F(s)|e^{j\varphi(s)} \qquad (11\text{-}2)$$

其中$|F(s)|$为复信号$F(s)$的模，而$\varphi(s)$为$F(s)$的相角。

从三维几何空间的角度来看，$|F(s)|$和$\varphi(s)$对应着复平面上的两个曲面，如果我们能绘出它们的三维曲面图，我们就可以直观地分析连续信号的拉普拉斯变换$F(s)$随复变量s的变化。

上述过程我们可以利用 MATLAB 的三维绘图功能来实现。现在考虑如何用 MATLAB 来绘制s平面的有限区域上连续时间信号$f(t)$的拉普拉斯变换$F(s)$的曲面图，我们以单位阶跃信号$\varepsilon(t)$为例来说明实现过程。

我们知道，对单位阶跃信号$f(t)=\varepsilon(t)$，其拉普拉斯变换为$F(s)=\dfrac{1}{s}$。

首先，我们用两个向量来确定绘制曲面图的s平面的横、纵坐标的范围。例如，我们可定义绘制曲面图的横坐标范围向量 x1 和纵坐标范围向量 y1 分别为：

```
x1=-0.2:0.03:0.2;
y1=-0.2:0.03:0.2;
```

然后再调用前面介绍过的 meshgrid()函数来产生矩阵 *S*，并用该矩阵来表示绘制曲面图的复平面区域，对应的 MATLAB 命令如下：

```
[x,y]=meshgrid(x1,y1);
s=x+i*y;
```

上述命令产生的矩阵 *s* 包含了复平面$-0.2<\sigma<0.2$、$-0.2<j\omega<0.2$范围内以间隔 0.03 取样的所有样点。

最后我们再计算出信号拉普拉斯变换在复平面的这些样点上的值，即可用函数 *mesh* 绘出其曲面图，对应命令为：



```
fs=abs(1./s);                              %计算拉氏变换在复平面上的样点值
mesh(x,y,fs);                              %绘制的拉氏变换曲面图
surf(x,y,fs);
title('单位阶跃信号拉氏变换曲面图');
colormap(hsv);
axis([-0.2,0.2,-0.2,0.2,0,60]);
rotate3d;
```

执行上述命令后，绘制的单位阶跃信号拉普拉斯曲变换面图如图 11-1 所示。

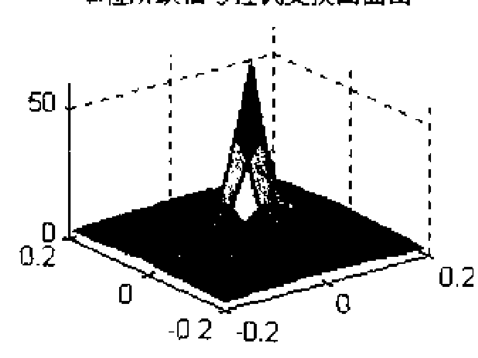


图 11-1　阶跃信号拉氏变换曲面图

例 11-1：已知连续时间信号 $f(t)=\sin(t)\varepsilon(t)$，求出该信号的拉普拉斯变换，并用 MATLAB 绘制拉普拉斯变换的曲面图。

解：

该信号的拉普拉斯变换为：

$$F(s)=\frac{1}{s^2+1}\qquad \mathrm{Re}(s)>0$$

我们可以用上面介绍的方法来绘出单边正弦信号的拉普拉斯变换的曲面图，实现这一过程的程序如下：

```
%绘制单边正弦信号拉普拉斯变换曲面图程序
clf;
a=-0.5:0.08:0.5;
b=-1.99:0.08:1.99;
[a,b]=meshgrid(a,b);
d=ones(size(a));
c=a+i*b;                                   %确定绘制曲面图的复平面区域
c=c.*c;
c=c+d;
c=1./c;
c=abs(c);                                  %计算拉普拉斯变换的样值
mesh(a,b,c);                               %绘制曲面图
```

```
surf(a,b,c);
axis([-0.5,0.5,-2,2,0,15]);
title('单边正弦信号拉氏变换曲面图');
colormap(hsv);
```

上述程序绘制的曲面图如图 11-2 所示。

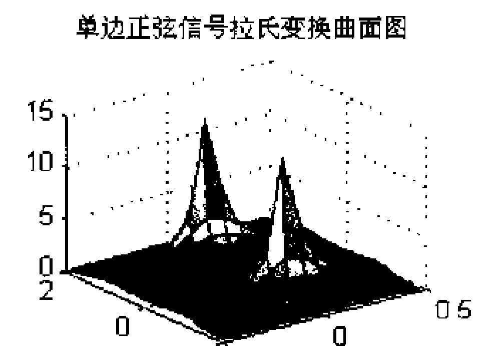


图 11-2　单边正弦信号拉氏变换曲面图

11.1.2　由拉普拉斯曲面图观察频域与复频域的关系

我们知道，若信号 $f(t)$ 的傅里叶变换存在，则其拉普拉斯变换 $F(s)$ 与傅里叶变换 $F(j\omega)$ 存在着如下关系：

$$F(j\omega)=F(s)\big|_{s=j\omega} \tag{11-3}$$

也即在信号拉普拉斯变换 $F(s)$ 中令 $\sigma=0$，就可得到信号的傅里叶变换。从三维几何空间的角度来看，信号 $f(t)$ 的傅里叶变换 $F(j\omega)$ 就是其拉普拉斯曲面图中虚轴（$\sigma=0$）所对应的曲线。我们可以通过将 $F(s)$ 曲面图在虚轴上进行剖面来直观地观察信号拉普拉斯变换与其傅里叶变换的对应关系。

例 11-2：试利用 MATLAB 绘制信号 $f(t)=\varepsilon(t)-\varepsilon(t-2)$ 的拉普拉斯变换的曲面图，观察曲面图在虚轴剖面上的曲线，并将其与信号傅里叶变换 $F(j\omega)$ 绘制的振幅频谱进行比较。

解：

根据拉普拉斯变换和傅里叶变换的定义和性质，我们可求得该信号的拉普拉斯变换和傅里叶变换如下：

$$F(s)=\frac{1-e^{-2s}}{s} \qquad\qquad F(j\omega)=2Sa(\omega)e^{-j\omega}$$

我们可用前面介绍的方法来绘制该信号的拉普拉斯变换曲面图。为了更好地观察曲面图在虚轴剖面上的曲线，我们定义制绘曲面图的 S 平面实轴范围从 0 开始，并用 view 函数来调整观察视角。实现上述过程的 MATLAB 程序如下：



```
%绘制矩形信号拉普拉斯变换曲面图程序
clf;
a=-0:0.1:5;
b=-20:0.1:20;
[a,b]=meshgrid(a,b);
c=a+i*b;                          %确定绘图区域
c=(1-exp(-2*c))./c;
c=abs(c);                         %计算拉普拉斯变换
mesh(a,b,c);                      %绘制曲面图
surf(a,b,c);
view(-60,20)                      %调整观察视角
axis([-0,5,-20,20,0,2]);
title('拉普拉斯变换（S 域像函数）');
colormap(hsv);
```

上述程序绘制的拉普拉斯变换的曲面图如图 11-3 所示。从该曲面图我们可以明显地观察到 $F(s)$ 在虚轴剖面上曲线的变化情况。

现在我们用 MATLAB 来绘制该信号的傅里叶变换曲线（振幅频谱），对应的 MATLAB 命令如下：

```
%绘制矩形时间信号傅里叶变换曲线程序
w=-20:0.1:20;                             %确定频率范围
Fw=(2*sin(w).*exp(i*w))./w;               %计算傅里叶变换
plot(w,abs(Fw))                           %绘制信号振幅频谱曲线
title('傅里叶变换（振幅频谱曲线）')
xlabel('频率 w')
```

上述命令绘制的信号傅里叶变换曲线如图 11-4 所示。通过对图 11-3 和 11-4 的观察，我们可直观地观察到拉普拉斯变换和傅里叶变换的对应关系。

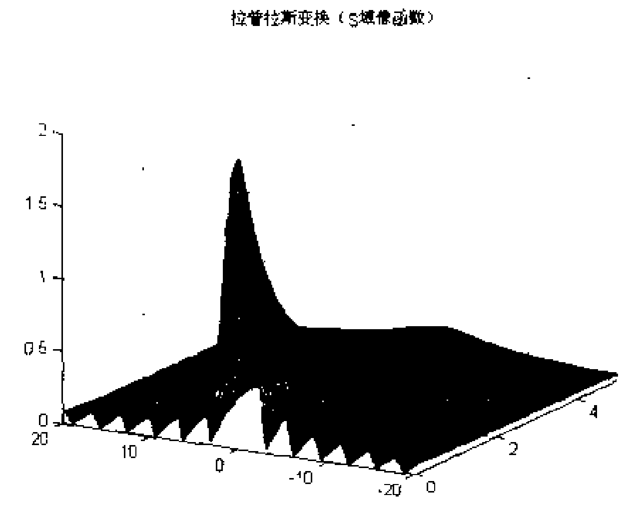


图 11-3　矩形信号拉氏变换曲面图

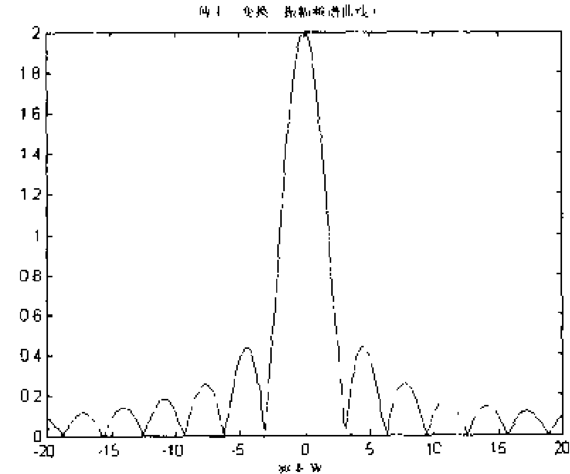


图 11-4　矩形信号傅里叶变换曲面图

11.1.3 拉普拉斯变换零极点分布对曲面图的影响

从单位阶跃信号和单边正弦信号的拉普拉斯变换曲面图我们可以看出，曲面图中均有突出的尖峰，仔细观察便可得出，这些峰值点在 S 平面的对应点就是信号拉普拉斯变换的极点位置。因此，如果将拉普拉斯变换曲面图比喻为一个地貌图的话，则极点位置就对应着山峰的峰点。

我们再来看拉普拉斯变换零点对曲面图的影响，考虑如下信号：

$$F(s)=\frac{2(s-3)(s+3)}{(s-5)(s^2+10)}$$

该信号的零点为 $q_{1,2}=\pm 3$，极点为 $p_{1,2}=\pm j3.1623$，$p_3=5$

我们用前面介绍的方法将其拉普拉斯变换的曲面图绘制出来，对应的 MATLAB 程序如下：

```
%观察拉普拉斯变换零极点对曲面图影响程序
clf;
a=-6:0.48:6;
b=-6:0.48:6;
[a,b]=meshgrid(a,b);
c=a+i*b;
d=2*(c-3).*(c+3);
e=(c.*c+10).*(c-5);
c=d./e;
c=abs(c);
mesh(a,b,c);
surf(a,b,c);
axis([-6,6,-6,6,0,3]);
title('拉普拉斯变换曲面图');
colormap(hsv);
view(-25,30)
```

从图 11-5 所示的曲面图我们可以明显看出，曲面图在 $s=\pm j3.1623$ 和 $s=5$ 处有三个峰点，对应着拉普拉斯变换的极点位置，而在 $s=\pm 3$ 处有两个谷点，对应着拉普拉斯变换的零点位置。因此，信号拉普拉斯变换的零极点位置，决定了其曲面图的峰点和谷点位置。



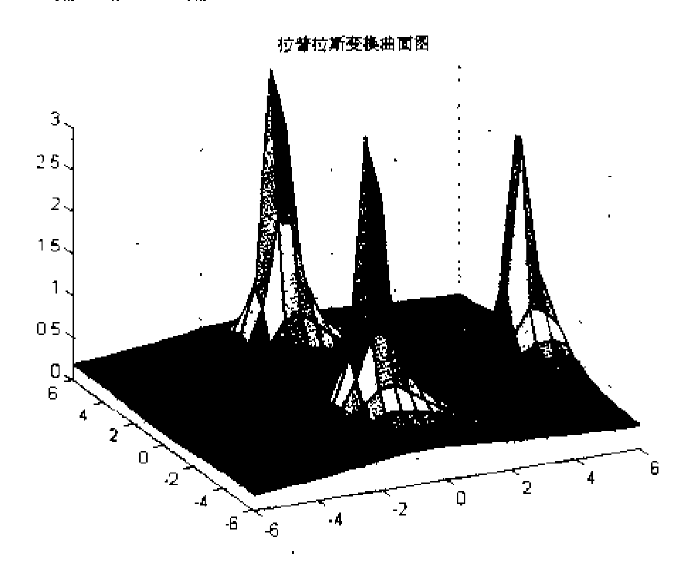


图 11-5 拉氏变换零极点分布曲面图

11.2 利用 MATLAB 绘制连续系统零极点图

线性时不变连续系统可以用如下所示的线性常系数微分方程来描述。

$$\sum_{i=0}^{N} a_i y^{(i)}(t) = \sum_{j=0}^{M} b_j f^{(j)}(t) \tag{11-4}$$

其中 $y(t)$ 为系统输出信号，$f(t)$ 为输入信号。

将上式进行拉普拉斯变换，则该连续系统的系统函数为：

$$H(s) = \frac{Y(s)}{F(s)} = \frac{\sum_{j=0}^{M} b_j s^j}{\sum_{i=0}^{N} a_i s^i} = \frac{B(s)}{A(s)} \tag{11-5}$$

式（11-5）中 $A(s)$ 和 $B(s)$ 分别是由微分方程系数决定的关于 S 的多项式，将上式因式分解后有：

$$H(s) = C\frac{\prod_{j=1}^{M}(s-q_j)}{\prod_{i=1}^{N}(s-p_i)} \tag{11-6}$$

其中 C 为常数，$q_j(j=1,2,\cdots,M)$ 为系统函数 $H(s)$ 的 M 个零点，$p_i(i=1,2,\cdots,N)$ 为 $H(s)$ 的 N 个极点。

可见，若连续系统的系统函数的零、极点已知，系统函数便可确定下来。即系统函数

$H(s)$ 的零、极点的分布完全决定了系统的特性。

因此，在连续系统的分析中，系统函数的零极点分布具有非常重要的意义。通过对系统函数零极点的分析，我们可以分析连续系统以下几个方面的特性：

- 系统冲激响应 $h(t)$ 的时域特性。
- 判断系统的稳定性。
- 分析系统的频率特性 $H(j\omega)$（幅频响应和相频响应）。

通过系统函数零极点分布来分析系统特性，首先就要求出系统函数的零极点，然后绘制零极点图。下面介绍如何利用 MATLAB 实现这一过程。

设连续系统的系统函数为：

$$H(s)=\frac{B(s)}{A(s)}$$

则系统函数的零点和极点位置可以用 MATLAB 的多项式求根函数 roots()来求得，调用函数 roots()的命令格式为：

```
p=roots(A)
```

其中 A 为待求根的关于 S 的多项式的系数构成的行向量，返回向量 p 则是包含该多项式所有根位置的列向量。例如多项式为：

$$A(s)=s^2+3s+4$$

则求该多项式根的 MATLAB 命令应为：

```
A=[1 3 4];
p=roots(A)
```

运行结果为：

```
p =
   -1.5000 + 1.3229i
   -1.5000 - 1.3229i
```

需要注意的是，系数向量 A 的元素一定要由多项式的最高幂次开始直到常数项，缺项要用 0 补齐。例如若多项式为：

$$A(s)=s^6+3s^4+2s^2+s-4$$

则表示该多项式的系数向量应为：

```
A=[1 0 3 0 2 1 -4]
```

用 roots()函数求得系统函数 $H(S)$的零极点后，就可以用 plot 命令在复平面上绘制出系统函数的零极点图，方法是在零点位置标以符号"X"，而在极点位置标以符号"O"。下面是求连续系统的系统函数零极点，并绘制其零极点图的 MATLAB 实用函数 sjdt()。

```
function [p,q]=sjdt(A,B)
%绘制连续系统零极点图程序
%A:系统函数分母多项式系数向量
%B:系统函数分子多项式系数向量
```



```
%p:函数返回的系统函数极点位置行向量
%q:函数返回的系统函数零点位置行向量
p=roots(A);                                    %求系统极点
q=roots(B);;                                   %求系统零点
p=p';                                          %将极点列向量转置为行向量
q=q';                                          %将零点列向量转置为行向量
x=max(abs([p q]));                             %确定纵坐标范围
x=x+0.1;
y=x;                                           %确定横坐标范围
clf
hold on
axis([-x x -y y]);                             %确定坐标轴显示范围
axis('square')
plot([-x x],[0 0])                             %画横坐标轴
plot([0 0],[-y y])                             %画纵坐标轴
plot(real(p),imag(p),'x')                      %画极点
plot(real(q),imag(q),'o')                      %画零点
title('连续系统零极点图')                        %标注标题
text(0.2,x-0.2,'虚轴')
text(y-0.2,0.2,'实轴')
```

例 11-3：已知连续系统的系统函数如下所示，试用 MATLAB 绘出系统的零极点图。

（1）$F(s)=\dfrac{s^2-4}{s^4+2s^3-3s^2+2s+1}$

（2）$F(s)=\dfrac{5s(s^2+4s+5)}{s^3+5s^2+16s+30}$

解：

我们可以直接调用 sjdt()函数来绘制上述系统的零极点图。对应的 MATLAB 命令如下：

```
a=[1 2 -3 2 1];
b=[1 -4];
sjdt(a,b)
a=[1 5 16 30];
b=[5 20 25 0];
sjdt(a,b)
```

上述命令绘制的系统零极点图如图 11-6 所示。



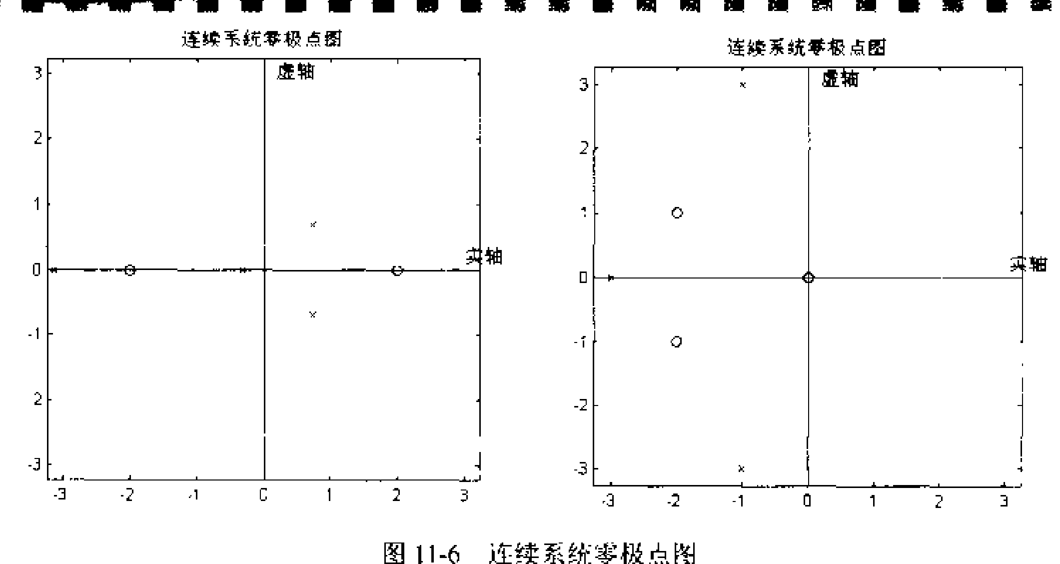


图 11-6 连续系统零极点图

11.3 连续系统零极点分析

11.3.1 零极点分布与系统稳定性

根据系统函数 $H(s)$ 的零极点分布来分析连续系统的稳定性是零极点分析的重要应用之一。稳定性是系统固有的性质，与激励信号无关，由于系统函数 $H(s)$ 包含了系统的所有固有特性，显然它也能反映出系统是否稳定。

对任意有界的激励信号 $f(t)$，若系统产生的零状态响应 $y(t)$ 也是有界的，则称该系统为稳定系统，否则，则称为不稳定系统。

可以证明，上述系统稳定性的定义可以等效为下列条件：

- 时域条件：连续系统稳定的充要条件为 $\int_{-\infty}^{\infty}|h(t)|\mathrm{d}t<\infty$，即系统冲激响应绝对可积。
- 复频域条件：连续系统稳定的充要条件为系统函数 $H(s)$ 的所有极点均位于 S 平面的左半平面内。

系统稳定的时域条件和复频域条件是等价的。因此，我们只要考察系统函数 $H(s)$ 的极点分布，就可判断系统的稳定性。对于三阶以下的低阶系统，我们可以利用求根公式方便地求出极点位置，从而判断系统的稳定性，但对于高阶系统，手工求解极点位置则显得非常困难。这时我们可以利用 MATLAB 来实现这一过程。

例 11-4：已知某连续系统的系统函数为：

$$H(s)=\frac{s^2+3s+2}{8s^4+2s^3+3s^2+s+5}$$

试用 MATLAB 求出该系统的零极点，画出零极点分布图，并判断系统是否稳定。



解：

调用前面介绍的绘制连续系统零极点图函数 sjdt 即可解决此问题，对应的 MATLAB 命令为：

```
a=[8 2 3 1 5];
b=[1 3 2];
[p,q]=sjdt(a,b)
```

运行结果为：

```
p =
   -0.6155 - 0.6674i   -0.6155 + 0.6674i    0.4905 - 0.7196i    0.4905 + 0.7196i
q =
-2      -1
```

绘制的系统零极图如图 11-7 所示。

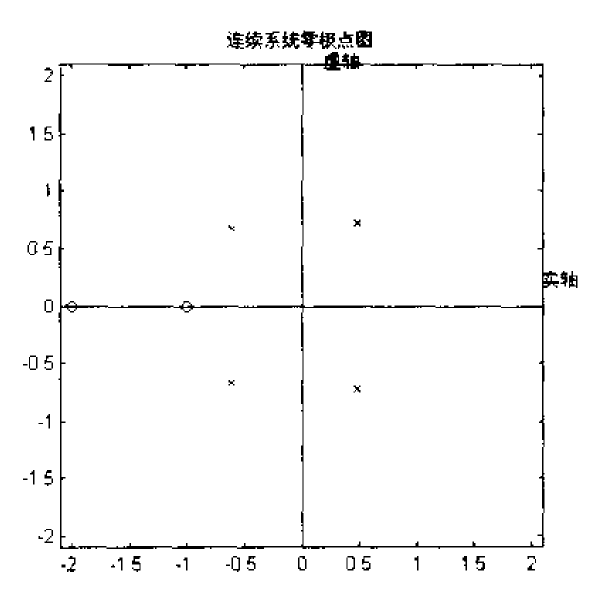


图 11-7 例 11-4 的系统零极点图

由程序运行结果可以看出，该系统在 S 平面的右半平面有一对共轭极点，故该系统是一个不稳定系统。

11.3.2 零极点分布与系统冲激响应时域特性

设连续系统的系统函数为 $H(s)$，冲激响应为 $h(t)$，则我们知道，$H(s)$ 与 $h(t)$ 是一对拉普拉斯变换对，即：

$$H(s)=\int_{-\infty}^{\infty} h(t)e^{-st}dt$$

显然，$H(s)$ 必然包含了 $h(t)$ 的本质特性。下面我们来分析 $H(s)$ 是如何决定 $h(t)$ 的时域特性的。

对于集中参数的 LTI 连续系统，其系统函数可表示为关于 S 的两个多项式之比，即：

$$H(s)=\frac{B(s)}{A(s)}=C\frac{\prod_{j=1}^{M}(s-q_j)}{\prod_{i=1}^{N}(s-p_i)} \tag{11-7}$$

其中$q_j(j=1,2,\cdots,M)$为$H(s)$的 M 个零点，$p_i(i=1,2,\cdots,N)$为$H(s)$的 N 个极点。

若系统函数的 N 个极点是单极点，则我们可将$H(s)$进行部分分式展开为：

$$H(s)=\sum_{i}^{N}\frac{k_i}{s-p_i} \tag{11-8}$$

从式（11-7）和（11-8）我们可以看出，系统冲激响应$h(t)$的时域特性完全由系统函数$H(s)$的极点位置决定。$H(s)$的每一个极点将决定$h(t)$的一项时间函数。显然$H(s)$的极点位置不同，则$h(t)$的时域特性也完全不同。

那么，$H(s)$的极点位置分布与$h(t)$的时域特性之间有何规律呢？我们用下面的例子来说明。

例 11-5：已知连续系统的零极点分布如图 11-8 所示，试用 MATLAB 分析系统冲激响应$h(t)$的时域特性。

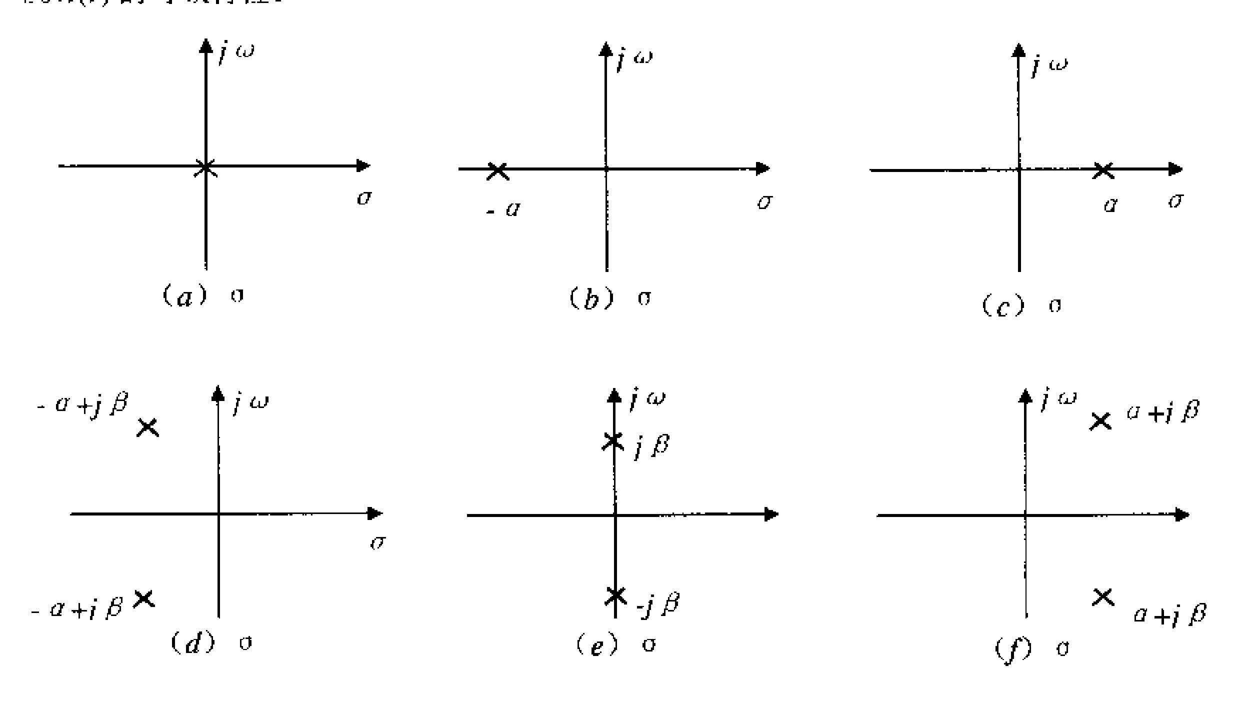


图 11-8 例 11-5 的系统零极点图

解：

系统的零极点图已知，则系统的系统函数$H(s)$就可确定。这样我们就可利用绘制连续系统冲激响应曲线的 MATLAB 函数 impulse()，将系统冲激响应$h(t)$的时域波形绘制出来。



对图 11-8（a）所示的系统，系统函数为 $H(s)=\frac{1}{s}$，即系统极点在原点，绘制冲激响应时域波形的 MATLAB 命令如下：

```
a=[1 0];
b=[1];
impulse(b,a)
```

绘制的冲激响应 $h(t)$ 的时域波形如图 11-9（a）所示，此时 $h(t)$ 为单位阶跃信号。

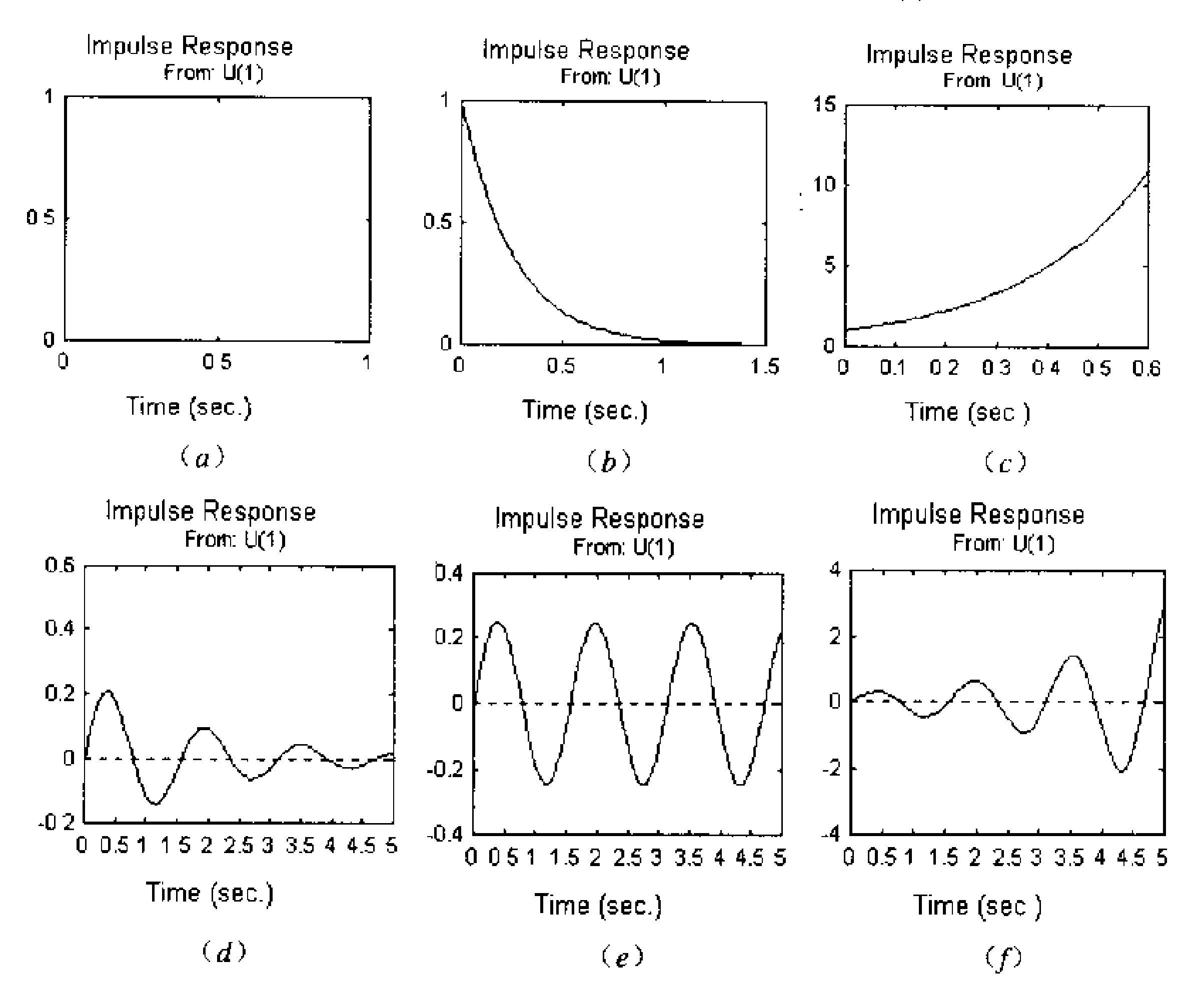


图 11-9 例 11-5 的系统冲激响应时域波形图

对图 11-8（b）所示的系统，系统函数为 $H(s)=\frac{1}{s+\alpha}$，即系统极点为位于 S 平面左半平面的实极点，令 α =2，绘制系统冲激响应时域波形的命令如下：

```
a=[1 2];
b=[1];
impulse(b,a)
```

绘制的系统冲激响应的时域波形如图 11-9（b）所示，此时 $h(t)$ 为衰减的指数信号。

对图 11-8（c）所示的系统，系统函数为 $H(s)=\dfrac{1}{s-\alpha}$，即系统极点为位于 S 平面右半平面的实极点，令 $\alpha=2$，绘制冲激响应时域波形的命令如下：

```
a=[1 -2];
b=[1];
impulse(b,a)
```

绘制的系统冲激响应的时域波形如图 11-9（c）所示，此时 $h(t)$ 为随时间增长的指数信号。

对图 11-8（d）所示的系统，系统函数为 $H(s)=\dfrac{1}{(s+\alpha)^2+\beta^2}$，即系统极点为位于 S 平面左半平面的一对共轭极点，令 $\alpha=0.5$、$\beta=4$，绘制冲激响应时域波形的命令如下：

```
a=[1 1 16.25];
b=[1];
impulse(b,a,5)
```

绘制的系统冲激响应的时域波形如图 11-9（d）所示，此时 $h(t)$ 为按指数规律衰减的正弦振荡信号。

对图 11-8（e）所示的系统，系统函数为 $H(s)=\dfrac{1}{s^2+\beta^2}$，即系统极点为位于 S 平面虚轴上一对共轭极点，令 $\beta=4$，绘制冲激响应时域波形的命令如下：

```
a=[1 0 16];
b=[1];
impluse(b,a,5)
```

绘制的系统冲激响应 $h(t)$ 的时域波形如图 11-9（e）所示，此时 $h(t)$ 为等幅正弦振荡信号。

对图 11-8（f）所示的系统，系统函数为 $H(s)=\dfrac{1}{(s-\alpha)^2+\beta^2}$，即系统极点为位于 S 平面右半平面的一对共轭极点，令 $\alpha=0.5$、$\beta=4$，绘制冲激响应时域波形的命令如下：

```
a=[1 -1 16.25];
b=[1];
impulse(b,a,5)
```

绘制的系统冲激响应 $h(t)$ 的时域波形如图 11-9（f）所示，此时 $h(t)$ 为按指数规律增长的正弦振荡信号。

从上述程序运行结果和绘制的系统冲激响应曲线，我们可以总结出以下规律：系统冲激响应 $h(t)$ 的时域特性完全由系统函数 $H(s)$ 的极点位置决定，$H(s)$ 位于 S 平面左半平面的极点决定了 $h(t)$ 随时间衰减的信号分量，位于 S 平面虚轴上的极点决定了冲激响应的稳态信号分量，位于 S 平面右半平面的极点决定了冲激响应随时间增长的信号分量。



11.3.3 由连续系统零极点分布分析系统的频率特性

由前面的分析可知，连续系统的零极点分布完全决定了系统的系统函数 $H(s)$，显然，系统的零极点分布也必然包含了系统的频率特性。

下面我们就介绍如何通过系统的零极点分布来直接求出系统频率响应 $H(j\omega)$ 的方法——几何矢量法，以及如何用 MATLAB 来实现这一过程。

几何矢量法是通过系统函数零极点分布来分析连续系统频率响应 $H(j\omega)$ 的一种直观而又简便的方法。该方法将系统函数的零极点视为 S 平面上的矢量，通过对这些矢量（零极点）的模和相角的分析，即可快速确定出系统的幅频响应和相频响应。其基本原理如下。

设某连续系统的系统函数为：

$$H(s)=\frac{B(s)}{A(s)}=\frac{\prod_{j=1}^{M}(s-q_j)}{\prod_{i=1}^{N}(s-p_i)}$$

其中，$q_j(j=1,2,\cdots,M)$ 和 $p_i(i=1,2,\cdots,N)$ 分别为系统函数的 M 个零点和 N 个极点，则系统的频率响应为：

$$H(j\omega)=H(s)\big|_{s=j\omega}=\frac{\prod_{j=1}^{M}(j\omega-q_j)}{\prod_{i=1}^{N}(j\omega-p_i)} \tag{11-9}$$

现在我们从几何矢量空间的角度来分析 S 平面，即将 S 平面的任一点看成是从原点到该点的矢量，则 $j\omega$ 即是从 S 平面原点到虚轴上角频率为 ω 的点的矢量。同理，$q_j(j=1,2,\cdots,M)$ 和 $p_i(i=1,2,\cdots,N)$ 即是原点到系统函数各零点和极点的矢量。

现考虑矢量 $j\omega-q_j$，由矢量运算可知，它实际上就是零点 q_j 到虚轴上角频率为 ω 的点的矢量，如图 11-10 所示。而矢量 $j\omega-p_i$ 就是极点 p_i 到虚轴上角频率为 ω 的点的矢量。

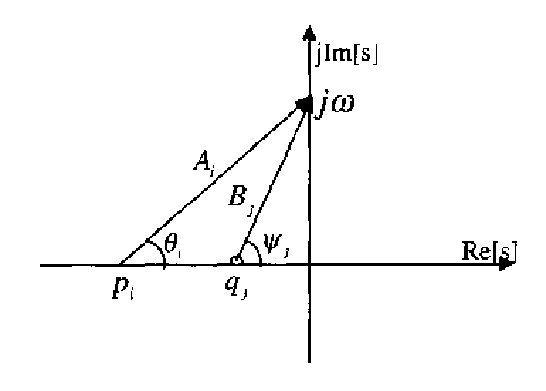


图 11-10 连续系统几何矢量法示意图

令：

$$j\omega - q_j = B_j e^{j\psi_j}$$

$$j\omega - p_i = A_i e^{j\theta_i}$$

则B_j就是零点q_j到虚轴上角频率为ω的点的矢量长度（距离），而ψ_j就是该矢量的相角。A_i就是极点p_i到虚轴上角频率为ω的点的矢量的长度（距离），而θ_i就是该矢量的相角。

因此有：

$$H(j\omega) = \frac{\prod_{j=1}^{M} B_j e^{j(\psi_1+\psi_2+\cdots+\psi_M)}}{\prod_{i=1}^{N} A_i e^{j(\theta_1+\theta_2+\cdots+\theta_N)}} = \left|H(e^{j\omega})\right| e^{j\varphi(\omega)} \tag{11-10}$$

则系统的幅频响应和相频响应为：

$$\left|H(j\omega)\right| = \frac{\prod_{j=1}^{M} B_j}{\prod_{i=1}^{N} A_i} \tag{11-11}$$

$$\varphi(\omega) = \sum_{j=1}^{M} \psi_j - \sum_{i=1}^{N} \theta_i \tag{11-12}$$

由上述分析我们可以得出以下结论：

- 连续系统的幅频响应$\left|H(j\omega)\right|$等于系统函数所有零点到虚轴上角频率为ω的点的距离之积与系统函数所有极点到虚轴上角频率为ω的点的距离之积的比值。
- 连续系统的相频响应$\varphi(\omega)$等于系统函数所有零点到虚轴上角频率为ω的点的矢量的相角之和与系统函数所有极点到虚轴上角频率为ω的点的矢量的相角之和的差值。

让矢量$j\omega$沿着虚轴变化，即角频率ω由$0\sim\infty$进行改变，我们便可直观地求出系统幅频响应和相频响应随ω的变化，从而分析出系统的频率特性。

根据上述结论，若已知系统的零极点分布，我们即可直接由几何矢量分析法分析出系统的频率特性。

上述过程我们可以用MATLAB来快速实现。用MATLAB实现已知系统零极点分布，求系统频率响应，并绘制其幅频特性和相频特性曲线的程序流程如下：

（1）定义包含系统所有零点和极点位置的行向量q和p。

（2）定义绘制系统频率响应曲线的频率范围向量$f1$和$f2$、频率取样间隔k（即频率变化步长值），并产生频率等分点向量f。

（3）求出系统所有零点和极点到这些等分点的距离。

（4）求出系统所有零点和极点到这些等分点的矢量的相角。



（5）根据式（11-11）和（11-12）求出 $f1$ 到 $f2$ 频率范围内各频率等分点的 $\left|H(e^{j\omega})\right|$ 和 $\varphi(\omega)$ 的值。

（6）绘制 $f1$～$f2$ 频率范围内系统的幅频特性曲线和相频特性曲线。

下面是实现上述分析过程的 MATLAB 实用函数 splxy()。

注意

在程序中尽可能地采用了矩阵运算而不是循环运算，例如流程中第（3）、（4）、（5）步的实现，这样可有效提高程序的运算速度。

```
function splxy(f1,f2,k,p,q)
%根据系统零极点分布绘制系统频率响应曲线程序
%f1、f2：绘制频率响应曲线的频率范围（即频率起始和终止点，单位为赫兹）
%p、q：系统函数极点和零点位置行向量
%k：绘制频率响应曲线的频率取样间隔
p=p';
q=q';
f=f1:k:f2;                              %定义绘制系统频率响应曲线的频率范围
w=f*(2*pi);
y=i*w;
n=length(p);
m=length(q);
if n==0                                      %如果系统无极点
    yq=ones(m,1)*y;
    vq=yq-q*ones(1,length(w));
    bj=abs(vq);
    ai=1;
elseif m==0                                %如果系统无零点
    yp=ones(n,1)*y;
    vp=yp-p*ones(1,length(w));
    ai=abs(vp);
    bj=1;
else
    yp=ones(n,1)*y;
    yq=ones(m,1)*y;
    vp=yp-p*ones(1,length(w));
    vq=yq-q*ones(1,length(w));
    ai=abs(vp);
    bj=abs(vq);
end
```

```
Hw=prod(bj,1)./prod(ai,1);
plot(f,Hw);
title('连续系统幅频响应曲线')
xlabel('频率 w（单位：赫兹）')
ylabel('F(jw)')
```

在上述程序中，若系统无零点或极点，则必须将零点或极点行向量定义为空向量。

下面我们举例来说明如何运用该实用程序来分析系统的频率特性。

例 11-6：已知某二阶系统的零极点分别为 $p_1=-\alpha_1$、$p_2=-\alpha_2$、$q_1=q_2=0$（二重零点），试用 MATLAB 分别绘出该系统在下列三种情况时，系统在 0~1kHz 频率范围内的幅频响应曲线，说明该系统的作用，并分析极点位置对系统频率响应的影响。

（1）$\alpha_1=100$，$\alpha_2=200$

（2）$\alpha_1=500$，$\alpha_2=1000$

（3）$\alpha_1=2000$，$\alpha_2=4000$

解：

这是个根据系统零极点分布来分析系统频率特性的典型问题，我们先调用函数 splxy() 来绘出上述三种情况下的系统幅频响应曲线，对应的 MATLAB 命令如下：

```
q=[0 0];
p=[-100 -200];
f1=0;
f2=1000;
k=0.1;
splxy(f1,f2,k,p,q)
p=[-500 -1000];
splxy(f1,f2,k,p,q)
p=[-2000 -4000];
splxy(f1,f2,k,p,q)
```

上述命令绘制的系统幅频响应曲线如图 11-11 所示。

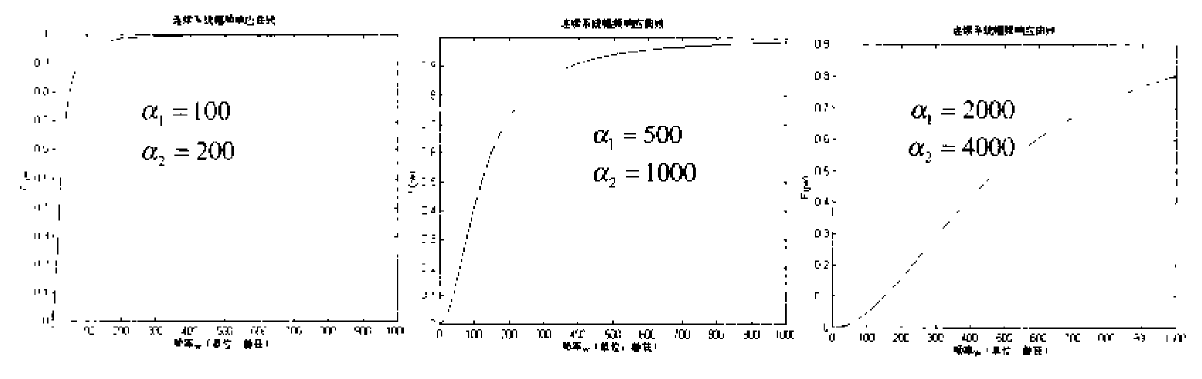
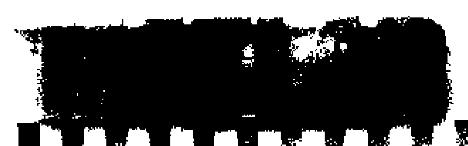


图 11-11　系统幅频响应曲线

由图 11-11 所示的系统幅频响应曲线我们可以看出，该系统呈高通特性，是一个二阶高通滤波器。当系统极点位置发生变化时，其高通特性也随之发生改变，当α_1、α_2离原



点较近时，高通滤波器的截止频率也较低，而当α_1、α_2离原点较远时，滤波器的截止频率也随之向高频方向移动。因此，我们就可以通过改变系统的极点位置来设计不同通带范围的高通滤波器。

例 11-7：已知连续系统的零极点分布分别如图 11-12 所示，试根据系统零极点分析的几何矢量分析法的原理，用 MATLAB 绘出系统的幅频响应曲线，并根据系统的幅频响应曲线分析系统的作用。

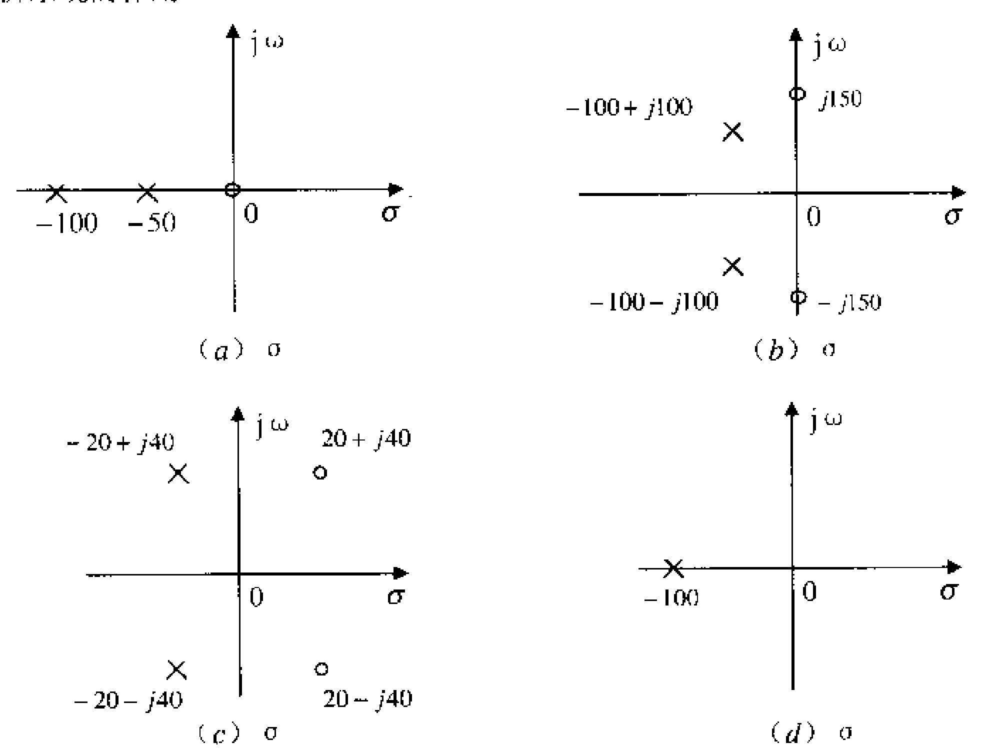


图 11-12　例 11-7 系统零极点图

对图 11-12（a）所示的系统，其零极点分别为$q_1 = 0$、$p_1 = -50$、$p_2 = -100$，我们可调用 splxy 函数来绘制系统的幅频响应曲线，对应的 MATLAB 命令如下：

```
q=[0];
p=[-50 -100];
f1=0;
f2=100;
k=0.1;
splxy(f1,f2,k,p,q)
```

绘制的系统幅频响应曲线如图 11-13（a）所示。由该系统的幅频响应曲线可以看出，该系统呈带通特性，是一带通滤波器。

对图 11-12（b）所示的系统，其零极点分别为$q_1 = j150$、$q_2 = -j150$、$p_1 = -100 + j100$、$p_2 = -100 - j100$，绘制系统的幅频响应曲线的 MATLAB 命令如下：

```
q=[i*150 -i*150];
```




```
p=[-100+i*100 -100-i*100];
f1=0;
f2=100;
k=0.1;
splxy(f1,f2,k,p,q)
```

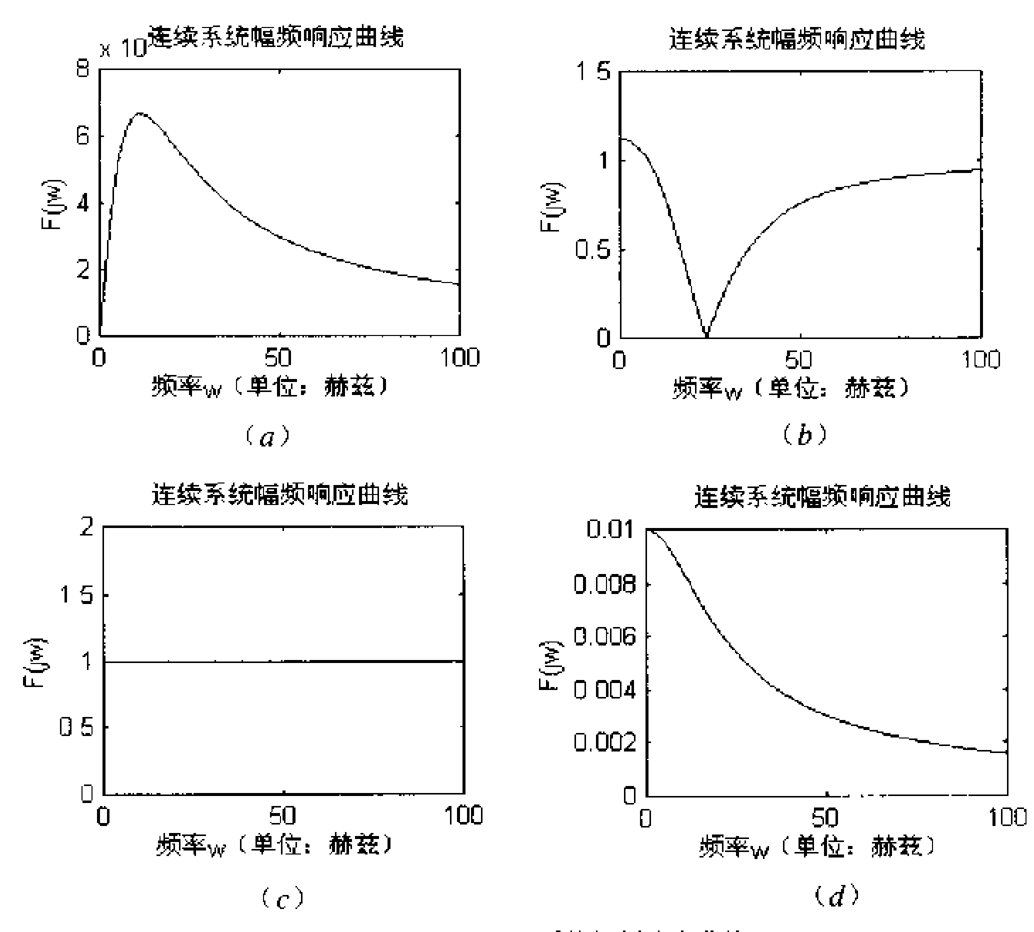


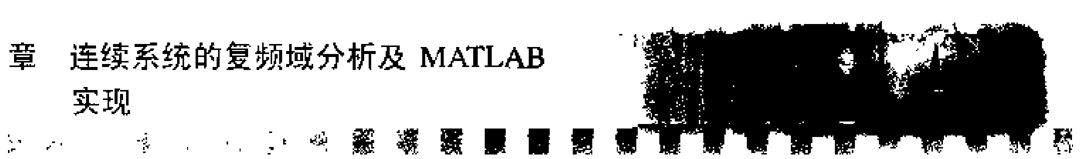
图 11-13　例 11-7 系统幅频响应曲线

绘制的系统幅频响应曲线如图 11-13（*b*）所示。由该系统的幅频响应曲线可以看出，该系统呈带阻特性，是一带阻滤波器。

对图 11-12（*c*）所示的系统，其零极点分别为 $q_1 = 20 + j40$ 、 $q_2 = 20 - j40$ 、 $p_1 = -20 + j40$ 、 $p_2 = -20 - j40$ ，绘制系统的幅频响应曲线的 MATLAB 命令如下：

```
q=[20+i*40 20-i*40];
p=[-20+i*40 -20-i*40];
f1=0;
f2=100;
k=0.1;
splxy(f1,f2,k,p,q)
```

绘制的系统幅频响应曲线如图 11-13（*c*）所示。由该系统的幅频响应曲线可以看出，该系统幅频响应为常数 1，是一全通滤波器。



对图 11-12（d）所示的系统，该系统无零点，只有极点分别为 $p_1=-100$，绘制系统的幅频响应曲线的 MATLAB 命令如下：

```
q=[ ];                                    %定义极点行向量为空向量
p=[-100];
f1=0;
f2=100;
k=0.1;
splxy(f1,f2,k,p,q)
```

绘制的系统幅频响应曲线如图 11-13（d）所示。由该系统的幅频响应曲线可以看出，该系统呈低通特性，是一低通滤波器。

11.4 巴特沃兹滤波器分析及 MATLAB 实现

理想低通滤波器的频率响应 $H(j\omega)$ 为：

$$H(j\omega)=\begin{cases}1 & \omega\leqslant\omega_c\\ 0 & \omega>\omega_c\end{cases}$$

其中 ω_c 称为低通滤波器的截止频率，$0\sim\omega_c$ 的频率范围称为滤波器的通带，$\omega_c\sim\infty$ 的频率范围称为滤波器的阻带。

对 $H(j\omega)$ 进行傅里叶逆变换可得，理想低通滤波器的冲激响应 $h(t)=\dfrac{\omega}{\pi}Sa(\dfrac{\omega_c t}{\pi})$。

从 $h(t)$ 的时域特性我们可以看出理想低通滤波器的一个非因果系统，因而是物理不可实现系统。但是在实际应用中，如果允许低通滤波器的通带和阻带之间有一定的过渡带，且通带和阻带允许有一定的衰减，我们就可以用物理可实现的系统去逼近理想低通滤波器的频率特性，从而获得较好的滤波效果。

巴特沃兹滤波器就是工程中常用的频率响应逼近理想低通滤波器的物理可实现系统。

巴特沃兹滤波器幅频响应 $|H(j\omega)|$ 的平方（模方函数）满足下式：

$$|H(j\omega)|^2=\frac{1}{1+(\dfrac{j\omega}{j\omega_c})^{2n}} \tag{11-13}$$

其中，n 称为巴特沃兹滤波器的阶数，ω_c 称为巴特沃兹滤波器的的截止频率。

下面我们用 MATLAB 来分析巴特沃兹滤波器频率特性。首先令 $\omega_c=100$，然后用 MATLAB 将巴特沃兹滤波器 n 取不同阶数时的频率响应曲线绘制出来。对应的 MATLAB 命令如下：

```
w=0:0.1:300;
wc=100;
```

```
for n=1:2:7
      hw=1./sqrt(1+(w/wc).^(2*n));
      hold on
      plot(w,hw)
end
title('巴特沃兹滤波器幅频响应曲线')
xlabel('角频率 w')
ylabel('H(jw)')
```

绘制的巴特沃兹滤波器的幅频响应如图 11-14 所示。

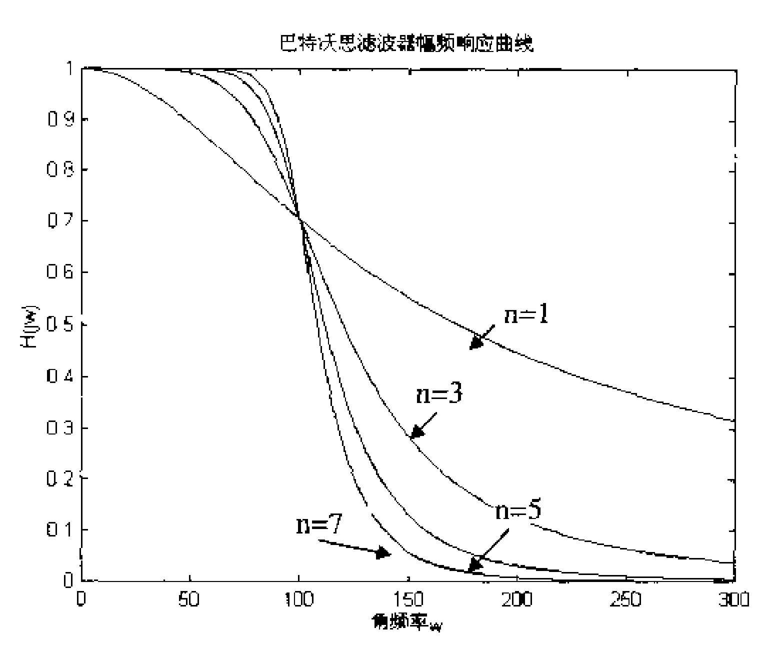


图 11-14　巴特沃兹滤波器频率特性曲线

从图 11-14 所示的系统幅频响应曲线我们可以看出，巴特沃兹滤波器的幅频响应曲线是单调变化的（即幅频响应曲线无起伏变化）。随着阶数 n 的增加，巴特沃兹滤波器的频率特性也逐渐向理想低通滤波器逼近。即当巴特沃兹滤波器的阶数较高时，我们就可以达到较为理想的滤波效果。

下面我们来分析巴特沃兹滤波器的系统函数及其零极点分布的规律。

由式（11-13）可得：

$$|H(j\omega)|^2 = H(j\omega)H(j\omega)^* = H(j\omega)H(-j\omega)$$

由于巴特沃滤波器是物理可实现的稳定因果系统，故有：

$$|H(s)|^2 = H(s)H(s)^* = H(s)H(-s) = \frac{1}{1+(\frac{s}{j\omega_c})^{2n}} \tag{11-14}$$

令：



$$1+(\frac{s}{j\omega_c})^{2n}=0$$

我们即可求出 $H(s)H(-s)$ 的 $2n$ 个极点 $p_i(i=1,2,\cdots\cdots,2n)$。由于巴特沃滤波器是稳定系统，故这 $2n$ 个极点中，位于左半平面的 n 个极点就是系统函数 $H(s)$ 的极点，而位于右半平面的 n 个极点则是 $H(-s)$ 的极点。

下面是求出巴特沃兹滤波器的极点位置并绘制其零极点分布图的 MATLAT 实用函数 batpq()。

```
function batpq(n,wc)
%绘制巴特沃兹滤波器零极点图程序
%n：巴特沃兹滤波器的阶数
%wc：巴特沃兹滤波器的截止频率
hold off
a=[1./((i*wc)^(2*n)) zeros(1,2*n-1) 1];  %定义系统函数分母多项式系数向量
b=[1];                                    %定义系统函数分子多项式
系数向量
p=roots(a)
sjdt(a,b)                                 %绘制系统零极点分布图
u=0:pi/200:2*pi;
r=wc*exp(i*u);
plot(r,':')                               %画半径为 wc 的圆
set(gcf,'color',[1 1 1])
set(gca,'box','on')
title('巴特沃兹滤波器极点分布图')
xlabel('S 平面实轴')
ylabel('S 平面虚轴')
```

注意

上述程序调用了绘制连续系统零极点图的实用函数 sjdt。我们运行如下 MATLAB 命令，即可求出巴特沃兹滤波器分别当 n=2、n=3、n=4 和 n=5 时的极点位置，并绘出其相应的零极点图。

```
for n=2:5
   n
   batpq(n,100)
end
```

程序运行结果如下：

```
n =

     2

p =
```

```
  -70.7107 +70.7107i
  -70.7107 -70.7107i
   70.7107 +70.7107i
   70.7107 -70.7107i
n =
     3
p =
   1.0e+002 *
   -1.0000
   -0.5000 + 0.8660i
   -0.5000 - 0.8660i
    0.5000 + 0.8660i
    0.5000 - 0.8660i
    1.0000
n =
     4
p =
  -92.3880 +38.2683i
  -92.3880 -38.2683i
  -38.2683 +92.3880i
  -38.2683 -92.3880i
   38.2683 +92.3880i
   38.2683 -92.3880i
   92.3880 +38.2683i
   92.3880 -38.2683i
n =
     5
p =
   1.0e+002 *
    1.0000
   -0.8090 + 0.5878i
   -0.8090 - 0.5878i
   -0.3090 + 0.9511i
   -0.3090 - 0.9511i
    0.3090 + 0.9511i
    0.3090 - 0.9511i
    0.8090 + 0.5878i
    0.8090 - 0.5878i
    1.0000
```



绘制的系统零点分布图如图 11-15 所示。

图 11-15　巴特沃兹滤波器零极点图

从上述程序绘制的零极点图我们可以看出，巴特沃滤波器的极点均匀地分布在半径为其截止频率 ω_c 的圆周上。且当 n 为奇数时，系统在实轴上有极点，当 n 为偶数时，系统在实轴上没有极点。

在实际应用中，我们可以根据提出的技术指标来设计巴特沃滤波器，通常的设计指标为：

- 截止频率 ω_c
- 过渡带频率范围 $\Delta\omega$
- 阻带最大增益 ε

上述设计指标如图 11-16 所示。由于巴特沃兹滤波器的频率响应是单调变化的，因此选择合适的阶数 n，使得当频率 $\omega > \omega_c + \Delta\omega$ 时，系统的幅频响应小于给定的阈值 ε。

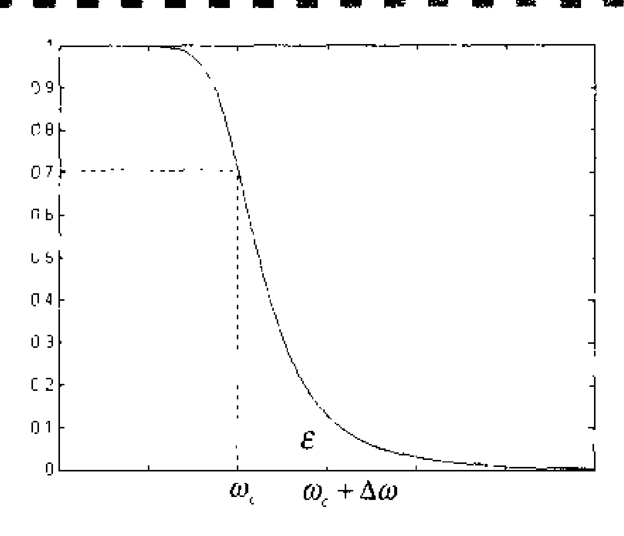


图 11-16　巴特沃兹滤波器设计示意图

对于二阶的情况，由式（11-14）可得，二阶巴特沃兹滤波器的系统函数为：

$$H(s)=\frac{\omega_c^{\ 2}}{s^2+\sqrt{2}\omega_c s+\omega_c}$$

11.5　拉普拉斯逆变换及 MATLAB 实现

连续信号 $f(t)$ 的拉普拉斯变换具有如下一般形式：

$$F(s)=\frac{C(s)}{D(s)}=\frac{\sum_{j=0}^{K}c_j s^j}{\sum_{i=1}^{L}d_i s^i}$$

若 $K\geqslant L$，则 $F(s)$ 可以分解为有理多项式与有理真分子之和，即：

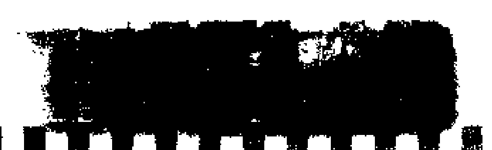
$$F(s)=P(s)+R(s)=P(s)+\frac{B(s)}{A(s)}=P(s)+\frac{\sum_{j=0}^{M}b_j s^j}{\sum_{i=1}^{N}a_i s^i}$$

其中，$P(s)$ 为关于 S 的多项式，其逆变换可直接求得（冲激信号及其各阶导数），$R(s)$ 为关于 S 的有理真分式，即满足 $M\leqslant N$。以下我们仅讨论 $M\leqslant N$ 的情况。

设连续信号 $f(t)$ 的拉普拉斯变换为 $F(s)$，则：

$$F(s)=\frac{B(s)}{A(s)}=\frac{B(s)}{\prod_{i=1}^{N}(s-p_i)}$$

其中 $p_i\,(i=1,2\cdots,N)$ 为 $F(s)$ 的 N 个极点。若满足 $M\leqslant N$，则可对其直接进行部分分



式展开得：

$$F(s)=\frac{r_1}{s-p_1}+\frac{r_2}{s-p_2}+\cdots+\frac{r_N}{s-p_N}$$

其中 $r_i=(s-p_i)F(s)\big|_{s=p_i}$　（ $i=1,2,\cdots N$ ）称为有理函数 $F(s)$ 的留数。

现分两种情况进行讨论。

11.5.1　$F(s)$ 的所有极点为单实极点

此时：

$$F(s)=\frac{r_1}{s-p_1}+\frac{r_2}{s-p_2}+\cdots+\frac{r_N}{s-p_N}$$

则 $F(s)$ 的拉普拉斯逆变换为：

$$f(t)=\sum_{i=1}^{N}r_i e^{p_i t}\varepsilon(t)$$

可见当 $F(s)$ 的所有极点为单实极点时，其对应时域信号 $f(t)$ 为若干个由 $F(s)$ 极点位置决定的指数信号之和。

11.5.2　$F(z)$ 有共轭极点

$$F(s)=\underbrace{\frac{r_1}{s-p_1}+\frac{r_2}{s-p}}_{f_2(t)}+\underbrace{\frac{r_3}{s-p_3}+\cdots+\frac{r_n}{s-p_n}}_{f_1(t)}$$

$F(s)\downarrow f(t)$

设 $F(s)$ 有一对有共轭极点 $p_{1,2}=-\alpha\pm j\beta$，则其中留数 $r_i(i=1,2,\cdots N)$ 的计算方法与 11.5.1 小节完全相同，即：

$$r_1=(s-p_1)F(s)\big|_{s-p_1}=|r_1|e^{\theta}$$

$$r_2=r_1^{*}$$

则 $F(s)$ 中由共轭极点 $p_{1,2}=-\alpha\pm j\beta$ 所决定的两项复指数信号可以合并为一项，故有：

$$f(t)=f_1(t)+f_2(t)=2|r_1|e^{-\alpha t}\cos(\beta t+\theta)\varepsilon(t)+\sum_{i=3}^{N}r_i e^{-p_i t}\varepsilon(t)$$

当$F(s)$具有一对以上共轭极点的情况时亦同理。

可见，当$F(s)$有共轭极点时，其对应时间信号将出现按指数规律变化的正弦（或余弦）振荡分量。

结论：连续时间信号$f(t)$的时域特性完全由其拉普拉斯变换$F(s)$的极点位置决定。

从以上的分析我们可以看出，只要求出$F(s)$部分分式展开的系数（留数）$r_i(i=1,2\cdots,N)$，我们就可以直接求出$F(s)$的逆变换$f(t)$。

上述求连续时间信号拉普拉斯逆变换的过程，我们可以用 MATLAB 的 residue 函数来实现。设：

$$F(s)=\frac{B(s)}{A(s)}=\frac{B(s)}{\prod_{i=1}^{N}(s-p_i)}=\sum_{i=1}^{N}\frac{r_i}{s-p_i}+\sum_{j=0}^{M-N}c_j s^j$$

若$M\leqslant N$，此项为零

令A和B分别是$F(s)$的分子和分母多项式构成的系数向量，则函数：

[r, p, k]=residue (B, A)

将产生三个向量r、p和k，其中p为包含$F(s)$所有极点的列向量，r为包含$F(s)$部分分式展开系数$r_i(i=1,2\cdots,N)$的列向量。k为包含$F(s)$部分分式展开的多项式项的系数$c_j(j=1,2\cdots,M-N)$的行向量，若$M\leqslant N$，则k为空阵。

用函数 residue()求出$F(s)$部分分式展开的系数后，便可根据其极点位置分布情况直接求出$F(s)$的拉普拉斯逆变换$f(t)$。下面我们举例说明如何用 MATLAB 求拉普拉斯逆变换。

例 11-8：已知连续信号的拉普拉斯变换为：

$$F(s)=\frac{2s+4}{s^3+4s}$$

试用 MATLAB 求其拉普拉斯逆变换$f(t)$。

解：

该问题可以用函数 residue()来解决，对应的 MATLAB 命令如下：

```
a=[1 0 4 0];
b=[2 4];
[r,p,k]=residue(b,a)
```

运行结果为：

```
r =
   -0.5000 - 0.5000i
   -0.5000 + 0.5000i
    1.0000
p =
         0 + 2.0000i
         0 - 2.0000i
         0
```



```
k =
  []
```

由上述结果我们可以看出，$F(s)$ 有三个极点 $p_{1,2}=\pm j2$、$p_3=0$，为了求得其轭极点对应的时间信号分量，我们可用 abs()和 angle()分别求出部分分式展开系数（留数）的模和相角，命令如下：

```
abs(r)
ans =
    0.7071
    0.7071
    1.0000
angle(r)/pi
ans =
   -0.7500
    0.7500
         0
```

由上述结果可得，$F(s)$ 的拉普拉斯逆变换为：$f(t)=[1+\sqrt{2}\cos(2t-\frac{3}{4}\pi)]\varepsilon(t)$

例 11-9：已知某连续系统的系统函数为：$H(s)=\dfrac{s+4}{s^3+3s^2+2s}$

试用 MATLAB 求出该系统的冲激响应 $h(t)$，并绘出其时域波形图，判断系统的稳定性。

解：

该问题可通过调用 residue()函数来解决，命令如下：

```
a=[1 3 2 0]
b=[1 4];
[r,p,k]=residue(b,a)
```

运行结果为：

```
r =
     1
    -3
     2
p =
    -2
    -1
     0
k =
  []
```

可见，系统函数有三个实极点，我们可以根据程序运行结果直接写出系统的冲激响应为：



$$h(t)=(e^{-2t}-3e^{-t}+2)\varepsilon(t)$$

由于系统的系统函数是已知的，故我们可直接调用 impulse()函数来绘制出其冲激响应 $h(t)$ 的时域波形，如图 11-17 所示。

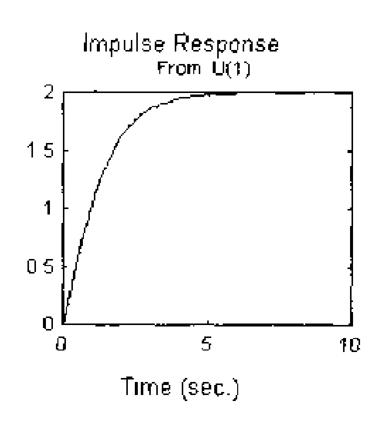


图 11-17　例 11-9 的系统冲激响应波形图

由时域波形可看出，当时间 t 趋于无穷大时，并不趋于零，故该系统是不稳定系统。另一方面，由于系统不满足所有极点都位于 S 平面的左半平的时域稳定条件，我们也可以得出相同的结论。

上机练习题

1．求下列信号的拉普拉斯变换，并用 MATLAB 绘制拉普拉斯变换在 S 平面的三维曲面图。

（1）　$f(t)=\cos(2t)\varepsilon(t)$

（2）　$f(t)=e^{-2t}\sin(t)\varepsilon(t)$

（3）　$f(t)=\sin(\pi t)[\varepsilon(t)-\varepsilon(t-1)]$

（4）　$f(t)=e^{-3t}\varepsilon(t)$

（5）　$f(t)=[1-e^{-2t}]\varepsilon(t)$

2．已知连续时间信号 $f(t)=\cos(2\pi t)[\varepsilon(t)-\varepsilon(t-4)]$，试求出该信号的拉普拉斯变换 $F(s)$ 及傅利变换 $F(j\omega)$，用 MATLAB 绘出该信号的拉普拉斯变换曲面图 $|F(s)|$ 及振幅频谱曲线 $|F(j\omega)|$，观察曲面图在虚轴上的剖面图，并将其与信号的振幅频谱曲线进行比较，分析频域与复频域的对应关系。

3．已知信号的拉普拉斯变换如下所示，试用 MATLAB 绘制其曲面图，观察拉普拉斯变换零极点分布对曲面图的影响。

（1）　$F(s)=\dfrac{1}{(s+2)(s+4)}$　　　　（2）　$F(s)=\dfrac{s}{(s+2)(s+4)}$



（3） $F(s)=\dfrac{(s+1)(s+4)}{s(s+2)(s+3)}$ （4） $F(s)=\dfrac{s^2-4}{s^2+4}$

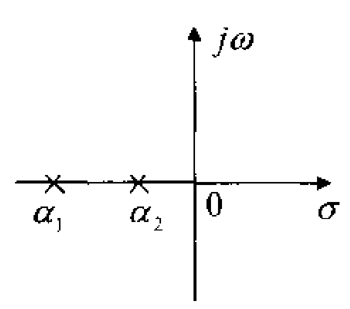
4．已知连续系统的系统函数分别如下所示，试用 MATLAB 绘制系统的零极点图，并根据零极点图判断系统的稳定性。

（1） $H(s)=\dfrac{s^2+s+2}{3s^3+5s^2+4s-6}$ （2） $H(s)=\dfrac{2(s^2-4s+5)}{s^2+4s+5}$

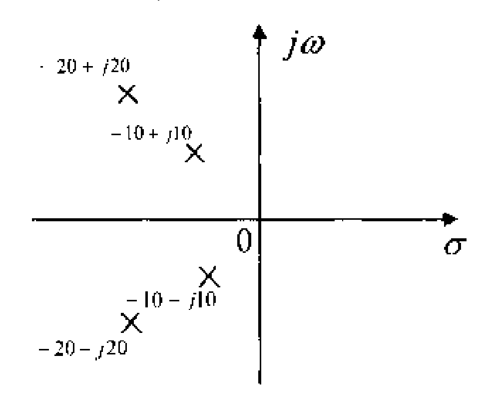
（3） $H(s)=\dfrac{3s(s^2-9)}{s^4+20s^2+64}$ （4） $H(s)=\dfrac{1}{s^3+2s^2+2s+1}$

5．对例 11-5 零极点图所示各系统，试改变（增大或减小）极点实部 α 和虚部 β 的大小，用 MATLAB 绘制 α 和 β 改变后系统冲激响应 $h(t)$ 的时域波形，观察 α 和 β 的大小是如何影响 $h(t)$ 的时域特性的，分析 α 和 β 的值的大小与时域特性的关系。当 $\alpha=0$ 时，$h(t)$ 有何特点？当 $\beta=0$ 时，$h(t)$ 又有何特点？

6．已知某二阶系统的零极点分布如下图所示，试根据系统零极点的几何矢量分析法的原理，用 MATLAB 绘出系统的幅频响应曲线，并用 MATLAB 指令编写绘制系统相频响应曲线的程序，绘出该系统的相频响应曲线，分析系统的作用。改变极点位置（改变 α_1 和 α_2 的大小），观察并分析极点位置是如何改变系统频率特性的。



7．已知某连续系统的零极点图如下图所示，试根据几何矢量分析法的原理，用 MATLAB 绘出该系统的幅频响应曲线，并分析该系统的作用。



8．请利用 MATLAB 设计一个巴特沃兹低通滤波器，具体设计指标为：

（1）系统截止频率 $\omega_c=1000\,\text{Hz}$

（2）过渡带范围 $\Delta f = 50\,\text{Hz}$

（3）阻带最大增益 $\varepsilon = 0.03$

试用 MATLAB 确定满足上述设计指标巴特沃兹低通滤波器的阶数 n，并绘出该滤波器的幅频响应曲线及零极点分布图。

9．试用 MATLAB 求下列信号的拉普拉斯逆变换。

（1）$F(s)=\dfrac{s+3}{s^2+3s+2}$　　（2）$F(s)=\dfrac{1}{s^3+2s^2+2s+1}$

（3）$F(s)=\dfrac{s^2+5s+4}{s^3+5s^2+6s}$　　（4）$F(s)=\dfrac{2s}{s^3+s+1}$

第12章　离散系统的Z域分析及MATLAB实现



12.1 利用 MATLAB 绘制离散系统零极点图

线性时不变离散系统可以用如下所示的线性常系数差分方程来描述。

$$\sum_{i=0}^{N} a_i y(k-i) = \sum_{j=0}^{M} b_j f(k-j) \tag{12-1}$$

其中 $y(k)$ 为系统输出序列，$f(k)$ 为输入系列。

将式（12-1）两边进行 Z 变换得：

$$H(z) = \frac{Y(z)}{F(z)} = \frac{\sum_{j=0}^{M} b_j z^{-j}}{\sum_{i=0}^{N} a_i z^{-i}} = \frac{B(z)}{A(z)} \tag{12-2}$$

式（12-1）中 $A(z)$ 和 $B(z)$ 分别是由描述系统的差分方程的系数决定的关于 z 的多项式，将式（12-2）因式分解后有：

$$H(z) = C\frac{\prod_{j=1}^{M}(z-q_j)}{\prod_{i=1}^{N}(z-p_i)} \tag{12-3}$$

其中 C 为常数，$q_j(j=1,2,\cdots,M)$ 为 $H(z)$ 的 M 个零点，$p_i(i=1,2,\cdots,N)$ 为 $H(z)$ 的 N 个极点。

由以上分析可以看出，系统函数 $H(z)$ 的零、极点的分布完全决定了系统的特性，若某离散系统的零点、极点已知，则系统函数便可确定下来。

因此，系统函数的零极点分布对我们进行离散系统特性的分析具有非常重要的意义。通过对系统函数零极点的分析，我们可以分析离散系统以下几个方面的特性：

- 系统单位响应 $h(k)$ 的时域特性
- 离散系统的稳定性
- 离散系统的频率特性（幅频响应和相频响应）

要通过系统函数零极点来分析系统特性，首先就要求出系统函数的零极点，然后绘制零点、极点图。MATLAB 为我们快速、高效地分析离散系统特性提供了强有力的工具，下面就介绍如何利用 MATLAB 实现这一过程。

设离散系统的系统函数为：

$$H(z) = \frac{B(z)}{A(z)}$$

则系统函数的零点和极点可以用 MATLAB 的多项式求根函数 roots（）来实现，调用





函数 roots（）的命令格式为：

```
p=roots(A)
```

其中 A 为待求根的多项式的系数构成的行向量，返回向量 p 则是包含该多项式所有根位置的列向量。例如多项式为：

$$B(z)=z^2+\frac{3}{4}z+\frac{1}{8}$$

则求该多项式根的 MATLAB 命令应为：

```
A=[1 3/4 1/8];
p=roots(A)
```

运行结果为：

```
p =
   -0.5000
   -0.2500
```

需要注意的是，在求系统函数零极点时，离散系统的系统函数可能有两种形式，一种是分子和分母多项式均按 Z 的降幂次序排列，如式（12-4）所示；另一种是分子多项式和分母多项式均按 Z^{-1} 的升幂次序排列，如式（12-5）所示。上述两种方式在构造多项式系数向量时稍有不同。

$$H(z)=\frac{z^3+2z}{z^4+3z^3+2z^2+2z+1} \tag{12-4}$$

$$H(z)=\frac{1+z^{-1}}{1+\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}} \tag{12-5}$$

若 $H(Z)$是以 Z 的降幂形式排列，则系数向量一定要由多项式的最高幂次开始，一直到常数项，缺项要用 0 补齐。例如对式（12-4）所示的系统函数，其分子多项式的系数向量应为：B=[1 0 2 0]（缺项用 0 补齐）

分母多项式的系数向量应为：A=[1 3 2 2 1]

若 $H(Z)$是以 Z^{-1} 的升幂形式排列，则分子和分母多项式系数向量的维数一定要相同，不足的要用 0 补齐，否则 Z=0 的零点或极点就可能被漏掉。例如，对式（12-5）所示的系统函数，其分子多项式系数向量应为：B=[1 1 0]（缺项用 0 补齐，以保证分子分母系数向量维数相同）

分母多项式系数向量应为：A=[1 1/2 1/4]

用 roots 函数求得 $H(Z)$的零极点后，就可以用 plot 命令绘制出系统函数的零极点图。下面是求系统函数零极点，并绘制其零极点图的 MATLAB 实用函数 ljdt()，该函数在绘出系统零极点图的同时，还绘出了 Z 平面的单位圆。

```
function ljdt(A,B)
% The function to draw the pole-zero diagram for discrete system
```




```
p=roots(A);                          %求系统极点
q=roots(B);                          %求系统零点
p=p';                                %将极点列向量转置为行向量
q=q';                                %将零点列向量转置为行向量
x=max(abs([p q 1]));                 %确定纵坐标范围
x=x+0.1;
y=x;                                 %确定横坐标范围
clf
hold on
axis([-x x -y y])                    %确定坐标轴显示范围
w=0:pi/300:2*pi;
t=exp(i*w);
plot(t)                              %画单位圆
axis('square')
plot([-x x],[0 0])                   %画横坐标轴
plot([0 0],[-y y])                   %画纵坐标轴
text(0.1,x,'jIm[z]')
text(y,1/10,'Re[z]')
plot(real(p),imag(p),'x')            %画极点
plot(real(q),imag(q),'o')            %画零点
title('pole-zero diagram for discrete system')   %标注标题
hold off
```

上述程序中，传入参量 A 和 B 分别是要绘制零极点图的系统函数的分母和分子多项式的系数向量。例如，现要分析三个离散系统，其系统函数分别如下：

（1）$H(z)=\dfrac{3z^3-5z^2+10z}{z^3-3z^2+7z-5}$

（2）$H(z)=\dfrac{1-0.5z^{-1}}{1+\frac{3}{4}z^{-1}+\frac{1}{8}z^{-2}}$

（3）$H(z)=\dfrac{-3z^{-1}}{2-5z^{-1}+2z^{-2}}$

则可直接运用上述绘图函数 ljdt()绘出三个离散系统的零极点分布图。调用绘制零极点图程序的命令及运行结果分别如下：

```
A=[1 -3 7 -5];
B=[3 -5 10 0];
ljdt(A,B)
```



绘制的零极点图如图 12-1 所示。

```
A=[1 3/4 1/8];
B=[1 -0.5 0];
ljdt(A,B)
```

运行结果如图 12-2 所示。

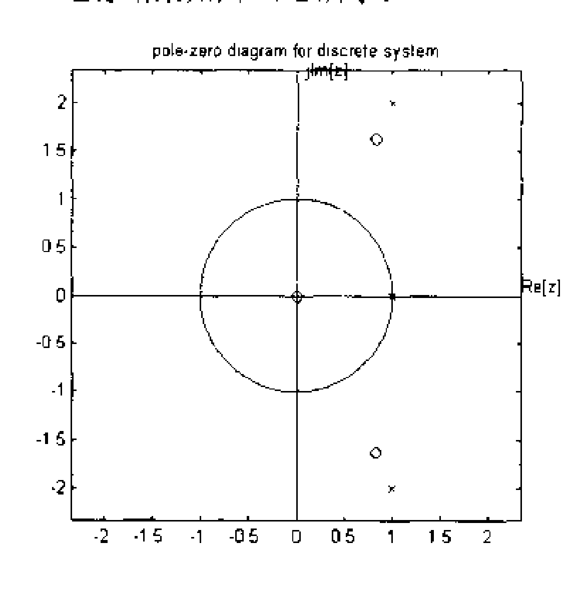


图 12-1　离散系统零极点图 1

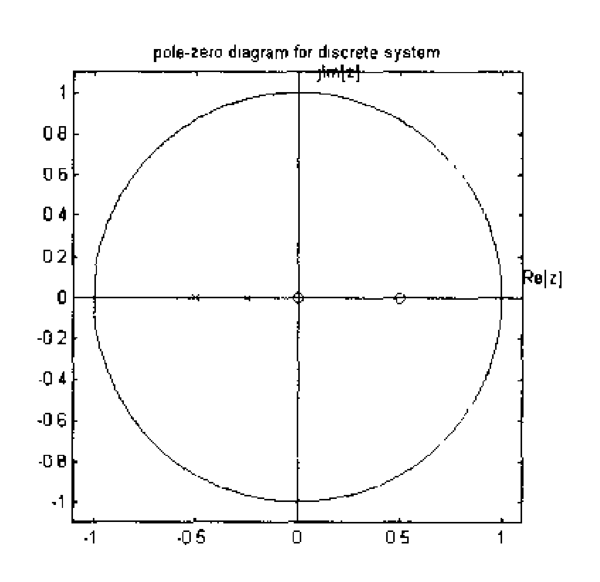


图 12-2　离散系统零极点图 2

```
A=[2 -5 2];
B=[0 -3 0];
ljdt(A,B)
```

运行结果如图 12-3 所示。

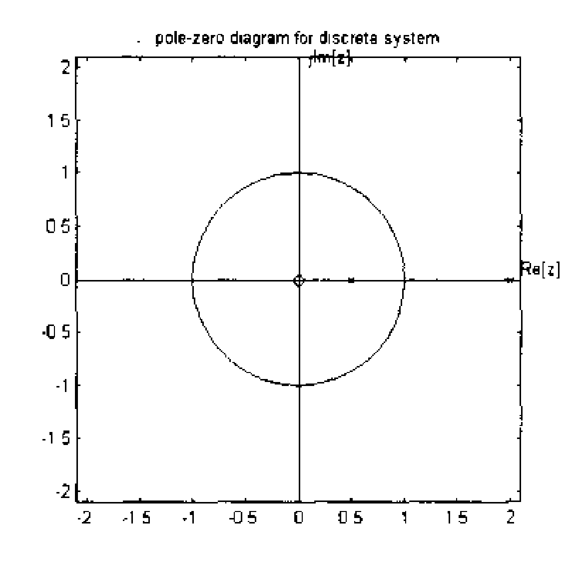


图 12-1　离散系统零极点图 3

12.2 离散系统的零极点分析

12.2.1 离散系统的零极点分布与系统稳定性

与连续系统的分析一样，根据系统函数 $H(z)$ 的零极点分布来分析离散系统的稳定性也是离散系统零极点分析的重要应用之一。

对任意有界的输入序列 $f(k)$，若系统产生的零状态响应 $y(k)$ 也是有界的，则称该离散系统为稳定系统，否则，则称为不稳定系统。

可以证明，上述系统稳定性的定义可以等效为下列条件：

- 时域条件：离散系统稳定的充要条件为 $\sum_{k=-\infty}^{\infty}|h(k)|<\infty$，即系统单位响应绝对求和。
- Z 域条件：离散系统稳定的充要条件为系统函数 $H(z)$ 的所有极点均位于 Z 平面的单位圆内。

离散系统稳定的时域条件和 Z 域条件是等价的。因此，我们只要考察系统函数 $H(z)$ 的极点分布，就可判断系统的稳定性。对于三阶以下的低阶系统，我们可以利用求根公式方便地求出离散系统的极点位置，从而判断系统的稳定性，但对于高阶系统，手工求解极点位置则显得非常困难。这时我们可以利用 MATLAB 来实现这一过程。

例 12-1：已知某离散系统的系统函数为：

$$H(z)=\frac{z+1}{3z^5-z^4+1}$$

试用 MATLAB 求出该系统的零极点，并画出零极点分布图，判断系统是否稳定。

解：

调用前面介绍的绘制离散系统零极点图函数 ljdt()即可解决此问题，对应的 MATLAB 命令为：

```
a=[3 -1 0 0 0 1];
b=[1 1];
ljdt(a,b)
p=roots(a)
q=roots(b)
pa=abs(p)
```

运行结果为：

```
p =
    0.7255 + 0.4633i
    0.7255 - 0.4633i
```



```
   -0.1861 + 0.7541i
   -0.1861 - 0.7541i
   -0.7455
q =
-1
pa =
     0.8608
     0.8608
     0.7768
     0.7768
     0.7455
```

绘制的系统零极图如图 12-4 所示。由程序运行结果和绘制的系统零极点图我们可以看出，该系统的所有极点均位于 Z 平面的单位圆内，故为稳定系统。

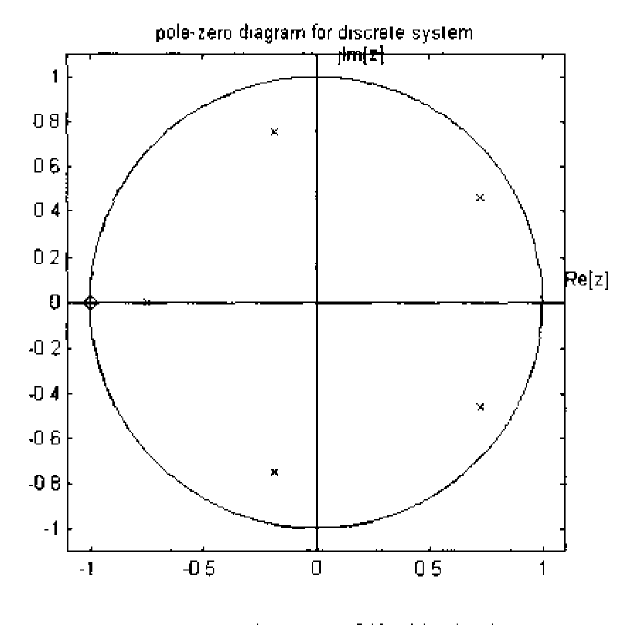


图 12-4　例 12-1 系统零极点图

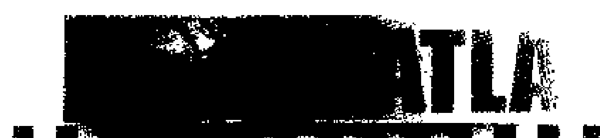
12.2.2　零极点分布与系统单位响应时域特性的关系

我们知道离散系统的系统函数 $H(z)$ 与其冲激响应 $h(k)$ 之间存在着如下关系：

$$H(z)=\sum_{k=-\infty}^{\infty}h(k)z^{-k}$$

即 $H(s)$ 与 $h(t)$ 是一对 Z 变换对。因而，$H(z)$ 必然包含了 $h(k)$ 的固有性质。下面我们来分析 $H(k)$ 是如何决定 $h(k)$ 的时域特性的。

离散系统的系统函数可表示为关于 Z 的两个多项式之比，即：



$$H(z)=\frac{B(z)}{A(z)}=C\frac{\prod_{j=1}^{M}(z-q_j)}{\prod_{i=1}^{N}(z-p_i)} \tag{12-6}$$

其中 $q_j(j=1,2,\cdots,M)$ 为 $H(z)$ 的 M 个零点，$p_i(i=1,2,\cdots,N)$ 为 $H(z)$ 的 N 个极点。

若系统函数的 N 个极点是单极点，我们可将 $H(z)$ 进行部分分式展开为：

$$H(z)=\sum_{i}^{N}\frac{k_i z}{z-p_i} \tag{12-7}$$

由 Z 逆变换可得：

$$h(k)=\sum_{i}^{N}k_i(p_i)^k\varepsilon(k) \tag{12-8}$$

从式（12-7）和（12-8）我们可以看出，离散系统单位响应 $h(k)$ 的时域特性完全由系统函数 $H(z)$ 的极点位置决定。$H(z)$ 的每一个极点将决定 $h(k)$ 的一项时间序列。显然 $H(z)$ 的极点位置不同，则 $h(k)$ 的时域特性也完全不同。

那么，$H(z)$ 的极点位置分布与 $h(k)$ 的时域特性之间有何规律呢？我们用下面的例子来说明。

例 12-2：已知离散系统的零极分布分别如图 12-5（a）、（b）、（c）、（d）、（e）、（f）、所示，其中虚线表示单位圆，试用 MATLAB 分析系统单位响应 $h(k)$ 的时域特性。

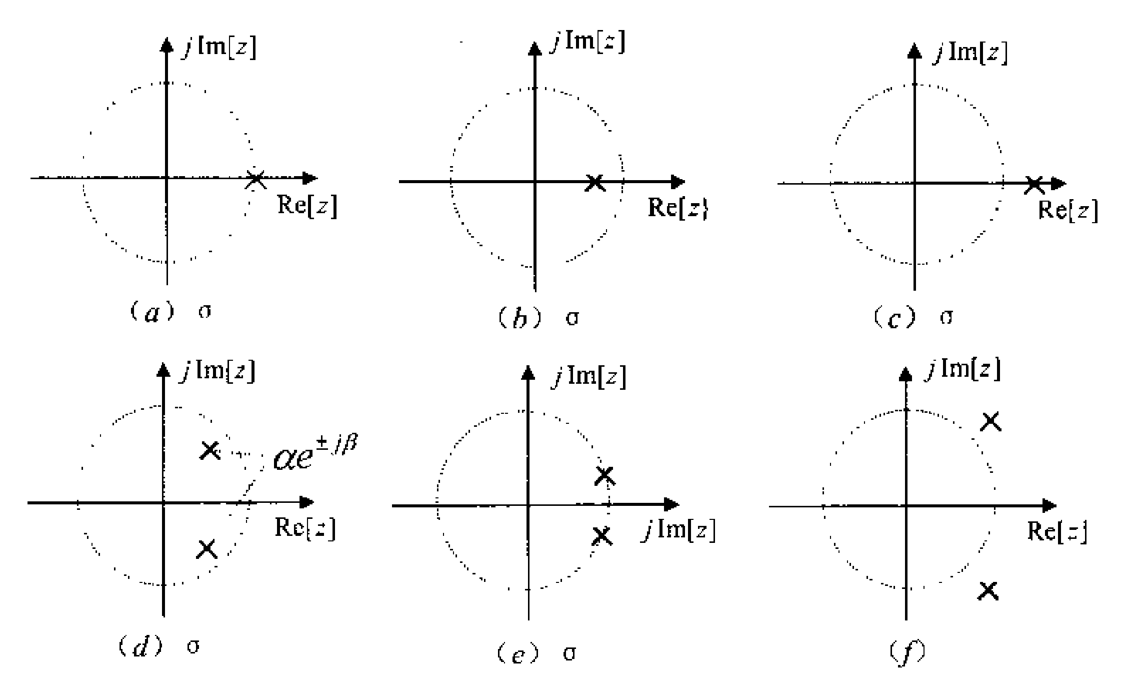


图 12-5　例 12-2 的系统零极点图



解：

系统的零极点图已知，则系统的系统函数 $H(z)$ 就可确定，这样我们就可利用绘制离散系统单位响应序列波形的 MATLAB 函数 impz，将上述不同极点分布情况下的系统单位响应 $h(k)$ 的序列波形绘制出来。

对图 12-5（a）所示的系统，系统函数为 $H(z)=\dfrac{1}{z-1}$，即系统极点为单位圆上实极点，则绘制单位响应时域波形的 MATLAB 命令如下：

```
a=[1 -1];
b=[1];
impz(b,a)
```

绘制的单位响应 $h(k)$ 的时域波形如图 12-6（a）所示，此时 $h(k)$ 为单位阶跃序列。

对图 12-5（b）所示的系统，系统函数为 $H(z)=\dfrac{1}{z-\alpha}$，其中 $\alpha<0$，即系统极点为位于 Z 平面单位圆内的实极点，令 $\alpha=0.8$，则绘制系统单位响应时域波形的命令如下：

```
a=[1 -0.8];
b=[1];
impz(b,a,10)
```

绘制的系统单位响应的时域波形如图 12-6（b）所示，此时 $h(k)$ 为衰减的指数序列。

对图 12-5（c）所示的系统，系统函数仍为 $H(z)=\dfrac{1}{z-\alpha}$，其中 $\alpha>0$，即系统极点为位于 Z 平面单位圆外的实极点，$\alpha=1.2$，则绘制系统单位响应时域波形的命令如下：

```
a=[1 -1.2];
b=[1];
impz(b,a,10)
```

绘制的系统单位响应的时域波形如图 12-6（c）所示，此时 $h(k)$ 为随时间增长的指数序列。

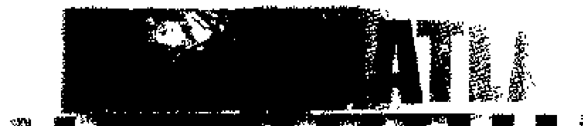
对图 12-5（d）所示的系统，系统函数为：

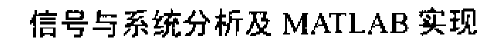
$$H(z)=\frac{1}{(z-\alpha e^{j\beta})(z-\alpha e^{-j\beta})}=\frac{1}{z^2-2\alpha\cos(\beta)z+\alpha^2}$$

其中 $\alpha<0$，即系统极点为位于 Z 平面单位圆内的一对共轭极点，取 $\alpha=0.8$、$\beta=\dfrac{\pi}{4}$，绘制单位响应时域波形的命令如下：

```
a=[1 -2*0.8*cos(pi/4) 0.8^2];
b=[1];
impz(b,a,20)
```

绘制的系统冲激响应的时域波形如图 12-6（d）所示，此时 $h(k)$ 为按指数规律衰减的





正弦振荡序列。

对图 12-5（e）所示的系统，系统函数为 $H(z)=\dfrac{1}{z^2-2\cos(\beta)z+1}$，即系统极点为位于 Z 平面单位圆上一对共轭极点，取 $\beta=\dfrac{\pi}{8}$ 绘制系统单位响应时域波形的命令如下：

```
a=[1 -2*cos(pi/8) 1];
b=[1];
impz(b,a,20)
```

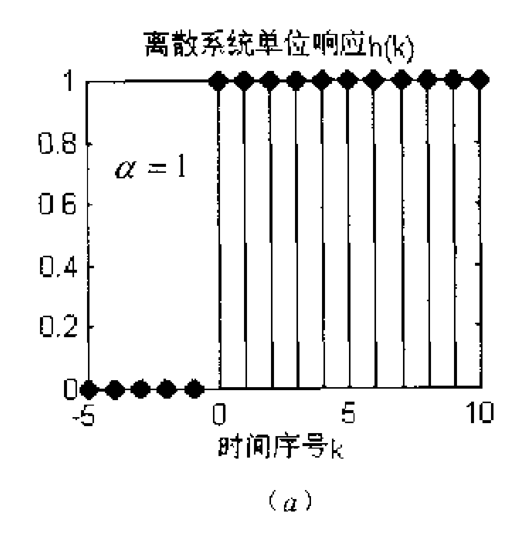
绘制的系统单位响应 $h(k)$ 的时域波形如图 12-6（e）所示，此时 $h(k)$ 为等幅正弦振荡序列。

对图 12-6（f）所示的系统，系统函数仍为 $H(z)=\dfrac{1}{z^2-2z\alpha\cos(\beta)+\alpha^2}$，其中 $\alpha>0$，即系统极点为位于 Z 平面单位圆外的一对共轭极点，取 $\alpha=1.2$、$\beta=\dfrac{\pi}{4}$，绘制系统单位响应时域波形的命令如下：

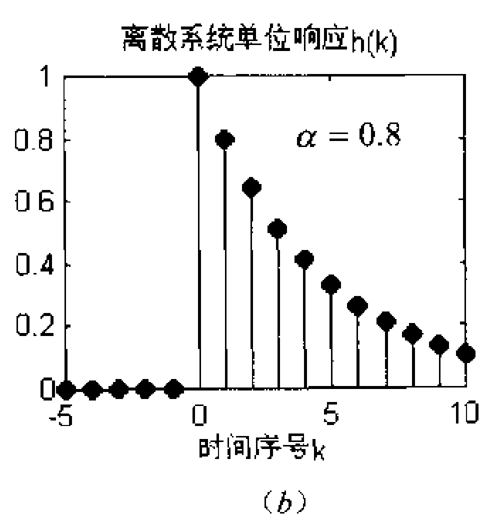
```
a=[1 -2*1.2*cos(pi/4) 1.2^2];
b=[1];
impz(b,a,20)
```

绘制的系统单位响应 $h(k)$ 的时域波形如图 12-6（f）所示，此时 $h(k)$ 为按指数规律增长的正弦振荡序列。

从上述程序运行结果和绘制的系统单位响应序列波形，我们可以总结出以下规律：离散系统单位响应 $h(k)$ 的时域特性完全由系统函数 $H(z)$ 的极点位置决定，$H(z)$ 位于 Z 平面单位圆内的极点决定了 $h(k)$ 随时间衰减的信号分量，位于 Z 平面单位圆上的极点决定了单位响应的稳定信号分量，位于 Z 平面单位圆外的极点决定了单位响应随时间增长的信号分量。



（a）



（b）



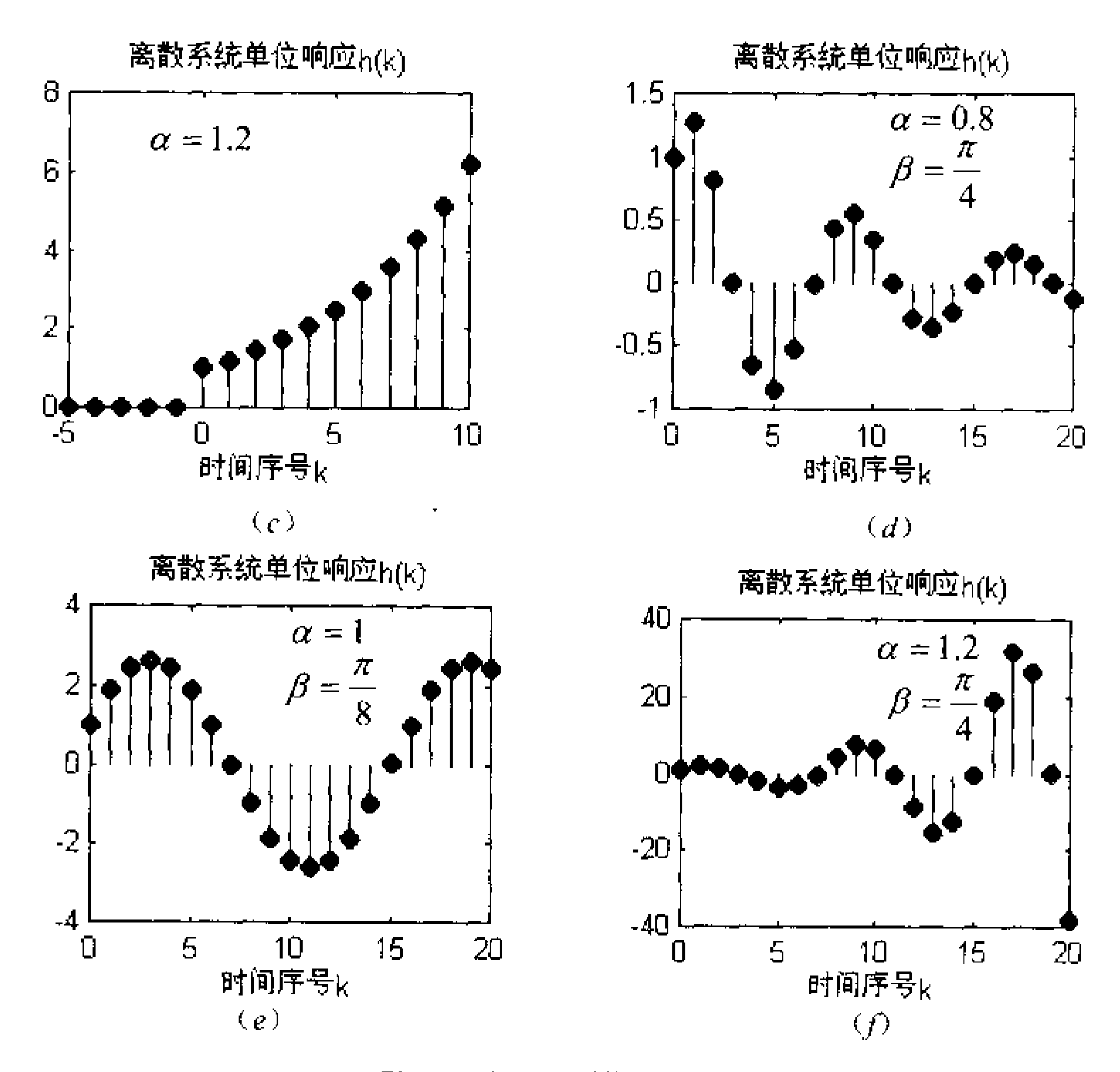


图 12-6　例 12-2 系统单位响应时域波形

12.3　离散系统的频率响应 $H(e^{j\omega})$

从以上的分析我们知道，离散系统的系统函数 $H(z)$反映了系统本身固有的特性。那么，当离散序列通过离散系统时，系统是如何对不同频率的输入序列进行加工和处理的呢？我们来分析一下这一过程。

设置某稳定的因果离散系统，其系统函数为$H(z)$，输入为正弦序列：

$$f(k)=A\sin(\omega k)\quad k\geqslant 0$$

该序列可视为由连续正弦时间信号 $f(t)=A\sin(\Omega t)$ 经过周期T 均匀抽样而得。则上式中 $\omega=\Omega T$ 称为离散正弦序列的数字角频率。

对输入正弦序列进行 Z 变换可得：

$$F(z)=\frac{Az\sin\omega}{z^2-2z\cos\omega+1}=\frac{Az\sin\omega}{(z-e^{j\omega})(z-e^{-j\omega})}$$

由离散系统的分析可知，设定系统的输出序列为 $y(k)$，则输出序列 $y(k)$ 的 Z 变换应为：

$$Y(z)=F(z)\cdot H(z)=\frac{Az\sin\omega}{(z-e^{j\omega})(z-e^{-j\omega})}\cdot H(z)$$

$$=\frac{Az\sin\omega}{(z-e^{j\omega})(z-e^{-j\omega})}\cdot\frac{B(z)}{\prod_{i=1}^{N}(z-p_i)}$$

上式中，$B(z)$ 为系统函数的分子多项式，$p_i(i=1,2,\cdots N)$ 为 $H(z)$ 的 N 个极点。用部分分式展开法对上式展开有：

$$Y(z)=\frac{k_1z}{z-e^{j\omega}}+\frac{k_2z}{z-e^{-j\omega}}+\sum_{i=1}^{N}\frac{A_iz}{z-p_i} \tag{12-9}$$

其中，$k_1,k_2,A_i(i=1,2,\cdots\cdots N)$ 为部分分式展开所确定的系数。显然，$e^{j\omega}$ 和 $e^{-j\omega}$ 可看做是 $Y(z)$ 的一对共轭极点。故：

$$k_1=\frac{Y(z)}{z}\cdot(z-e^{j\omega})\Big|_{z=e^{j\omega}}=\frac{A\sin\omega}{z-e^{j\omega}}\cdot H(z)\Big|_{z=e^{j\omega}}=\frac{A}{2j}\cdot H(e^{j\omega})$$

$$k_2=k_1^*=\frac{A}{-2j}\cdot H(e^{-j\omega})$$

令：$H(e^{j\omega})=\left|H(e^{j\omega})\right|\cdot e^{j\varphi(\omega)}$

则 $H(e^{-j\omega})=\left|H(e^{j\omega})\right|\cdot e^{-j\varphi(\omega)}$

代入（12-9）式有：

$$Y(z)=\frac{A}{2j}\left|H(e^{j\omega})\right|\cdot(\frac{ze^{j\varphi(\omega)}}{z-e^{j\omega}}-\frac{ze^{-j\varphi(\omega)}}{z-e^{-j\omega}})+\sum_{i=1}^{N}\frac{A_iz}{z-p_i}$$

对上式求逆变换有：

$$y(k)=A\left|H(e^{j\omega})\right|\sin[\omega k+\varphi(\omega)]+\sum_{i=1}^{N}A_ip^k{}_i \qquad k\ge 0$$

由于系统是稳定系统，故系统函数的所极点应在单位圆内，即：

$$\left|p_i\right|<1(i=1,2,\cdots N)$$



所以，系统的稳态响应为：

$$y(k)=A\left|H(e^{j\omega})\right|\sin[\omega k+\varphi(\omega)] \quad (12\text{-}10)$$

从式（12-10）我们可得出如下结论：离散系统对数字角频率为 ω 的正弦输入序列的处理，表现在幅度和相位两方面的改变上，$H(e^{j\omega})$ 的模决定了输出序列与输入序列的幅度之比，而 $H(e^{j\omega})$ 的相角则决定了输出序列和输入序列的相位之差。也就是说，$H(e^{j\omega})$ 的作用完全类似于连续系统的频率响应 $H(j\omega)$。因此，我们将：

$$H(e^{j\omega})=H(z)\big|_{z=e^{j\omega}}=\left|H(e^{j\omega})\right|e^{j\varphi(\omega)}$$

定义为离散系统的频率响应。$\left|H(e^{j\omega})\right|$ 称为离散系统的幅频响应，$\varphi(\omega)$ 称为离散系统的相频响应，$\left|H(e^{j\omega})\right|$ 随 ω 而变化的曲线称为系统的幅频特性曲线，$\varphi(\omega)$ 随 ω 而变化的曲线称为系统的相频特性曲线。

$H(e^{j\omega})$ 与连续系统的频率响应 $H(j\omega)$ 最大区别在于其呈周期性，且周期为 2π。因此，只要分析 $H(e^{j\omega})$ 在 $|\omega|\leqslant 2\pi$ 范围内的情况，便可分析出系统的整个频率特性。

在 $H(e^{j\omega})$ 中，由于 ω 是数字角频率，它与采样前连续信号的角频率的关系是：

$$\omega=\Omega T$$

其中 T 为采样周期。因此对于满足抽样定理的抽样情况，采样角频率 ω_s 应满足：

$$\omega_s=\frac{2\pi}{T}\geqslant 2\pi$$

即：

$$\omega=\Omega T\leqslant\pi$$

因此，在 $H(e^{j\omega})$ 随 ω 的变化关系中，$\omega=\pi$ 附近，反映了系统对输入信号高频部分的处理情况，而 $\omega=0$ 附近，则反映了系统对输入信号低频部分的处理情况。如图 12-7 所示的系统就是一个理想的数字低通滤波器，而图 12-8 所示的系统则是一个理想的数字高通滤波器。

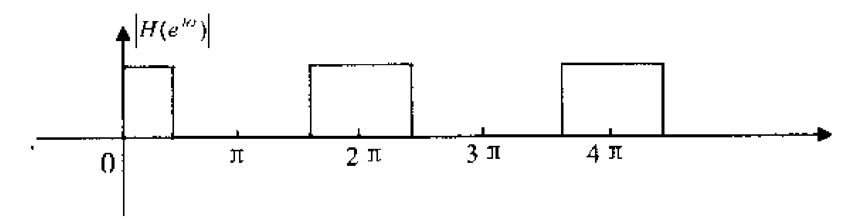


图 12-7 理想数字低通滤波器

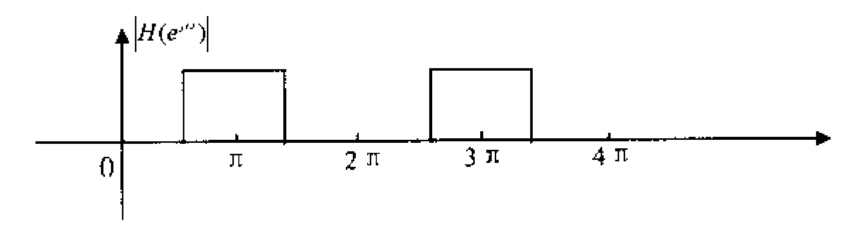
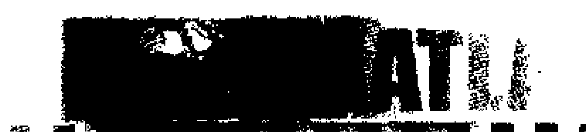


图 12-8 理想数字高通滤波器



12.4 用 MATLAB 实现离散系统的频率特性分析

由上述分析可知，离散系统的幅频特性曲线和相频特性曲线直观地反映了系统对不同频率的输入序列的处理情况。因此，我们只要知道离散系统的频率响应 $H(e^{j\omega})$，就可分析离散系统的整个频率特性。那么，如何求得离散系统的频率响应 $H(e^{j\omega})$ 呢？最简便的方法就是通过系统函数 $H(z)$ 的分析而得到系统的频率响应 $H(e^{j\omega})$，通常采用如下两种分析方法。

12.4.1 直接法

设某离散系统的系统函数为 $H(z)$，则该系统的频率响应为：

$$H(e^{j\omega}) = \left|H(e^{j\omega})\right| \cdot e^{j\varphi(\omega)} = H(z)\Big|_{z=e^{j\omega}}$$

MATLAB 为用户提供了专门用于求离散系统频率响应的函数 freqz ( )，调用 freqz ( ) 函数有如下两种格式。

1、[H，w]=freqz(B,A, N)

在上述调用中，B 和 A 分别是待分析的离散系统的系统函数分子、分母多项式的系数向量，N 为正整数，返回向量 H 则包含了离散系统频率响应 $H(e^{j\omega})$ 在 0~π 范围内 N 个频率等分点的值，向量 w 则包含 0~π 范围内的 N 个频率等分点。调用中若 N 默认，则系统默认为 N=512。

例如，对如下离散系统：

$$H(z) = \frac{z-0.5}{z} \tag{12-11}$$

则计算其 0~π 频率范围内 10 个频率等分点的频率响应 $H(e^{j\omega})$ 的样值的 MATLAB 命令为：

```
A =[1 0];
B =[1 -0.5];
[H,w]=freqz(B,A, 10)
```

运行结果为：

```
H =
   0.5000
   0.5245 + 0.1545i
   0.5955 + 0.2939i
   0.7061 + 0.4045i
   0.8455 + 0.4755i
   1.0000 + 0.5000i
   1.1545 + 0.4755i
```



```
    1.2939 + 0.4045i
    1.4045 + 0.2939i
    1.4755 + 0.1545i
w =
         0
    0.3142
    0.6283
    0.9425
    1.2566
    1.5708
    1.8850
    2.1991
    2.5133
    2.8274
```

2、[H，w]=freqz(B,A, N,'whole')

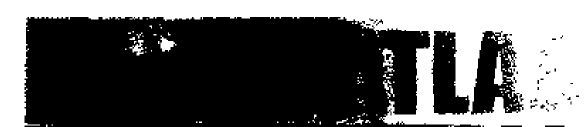
该调用格式将计算离散系统在 0~2π 范围内 N 个频率等分点的频率响应 $H(e^{j\omega})$ 的值。

因此，我们可以先调用 freqz()函数计算出离散系统频率响应的值，然后再利用 MATLAB 的 abs()和 angle()函数及 plot 命令，即可绘制出系统在 0~π 或 0~2π 范围内的幅频特性和相频特性曲线。例如，对于式（12-11）所示系统，绘制系统幅频特性和相频特性曲线的 MATLAB 命令如下：

```
B=[1 -0.5];
 A =[1 0];
 [H,w]=freqz(B,A,400,'whole');
 Hf=abs(H);
 Hx=angle(H);
 clf
 figure(1)
 plot(w,Hf)
 title('离散系统幅频特性曲线')
 figure(2)
 plot(w,Hx)
 title('离散系统相频特性曲线')
```

该程序绘制的系统频率特性曲线如图 12-9 和图 12-10 所示。

从该系统的幅频特性曲线可以看出，该系统呈高通特性，是一阶高通滤波器。



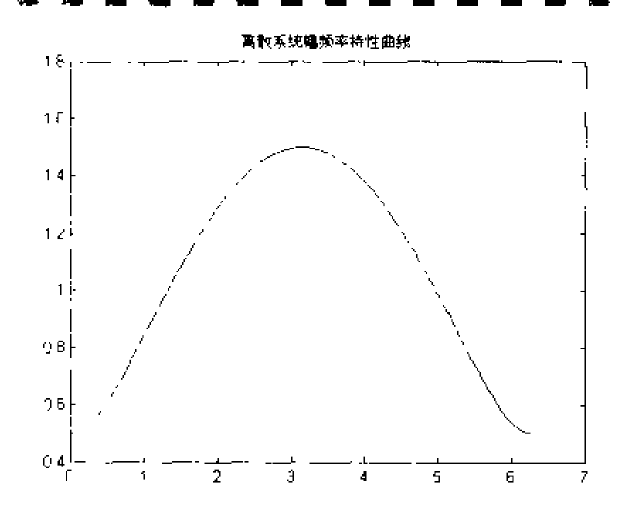


图 12-9　离散系统幅频特性曲线图

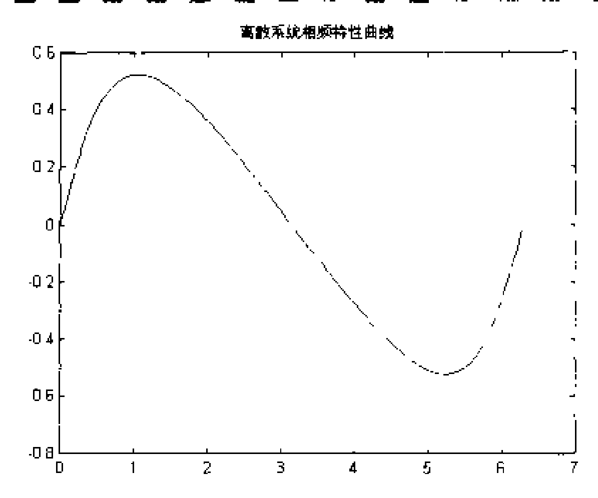


图 12-10　离散系统相频特性曲线

12.4.2　几何矢量法

几何矢量法是通过系统函数零极点分布来分析离散系统频率响应的一种直观而又简便的方法。该方法将系统函数的零极点视为 Z 平面上的矢量，通过对这些矢量（零极点）的模和相角的分析，即可快速确定出系统的幅频响应和相频响应。其基本原理如下。

设某离散系统的系统函数为：

$$H(e^{j\omega})=\frac{\prod_{j=1}^{M}(e^{j\omega}-q_j)}{\prod_{i=1}^{N}(e^{j\omega}-p_i)}$$

其中，$q_j(j=1,2,\cdots,M)$ 和 $p_i(i=1,2,\cdots,N)$ 分别为系统函数的 M 个零点和 N 个极点，则系统的频率响应为：

$$H(z)=\frac{B(z)}{A(z)}=\frac{\prod_{j=1}^{M}(z-q_j)}{\prod_{i=1}^{N}(z-p_i)}$$

从 Z 平面的矢量几何角度来考虑，可将 Z 平面的任一点看成是从原点到该点的矢量，则 $e^{j\omega}$ 即是从原点到单位圆的矢量（因其模恒为 1），ω 即是矢量 $e^{j\omega}$ 与 Z 平面实坐标轴的夹角。同理 $q_j(j=1,2,\cdots,M)$ 和 $p_i(i=1,2,\cdots,N)$ 即是原点到系统函数各零点和极点的矢量。

现考虑矢量 $e^{j\omega}-q_j$，由矢量运算可知，它实际上就是零点 q_j 到单位圆上的点 $e^{j\omega}$ 的矢量，如图 12-11 所示。



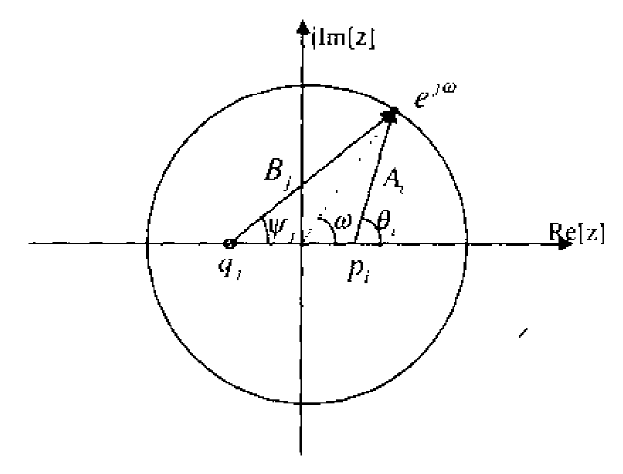


图 12-11　几何矢量法示意图

而矢量 $e^{j\omega}-p_i$ 就是极点 p_i 到单位圆上的点 $e^{j\omega}$ 的矢量。

令：

$$e^{j\omega}-q_j=B_je^{j\psi_j}$$

$$e^{j\omega}-p_i=A_ie^{j\theta_i}$$

则 B_j 就是零点 q_j 到单位圆上的点 $e^{j\omega}$ 的矢量的长度（距离），而 ψ_j 就是该矢量的相角。A_i 就是极点 p_i 到单位圆上的点 $e^{j\omega}$ 的矢量的长度（距离），而 θ_i 就是该矢量的相角。

因此有：

$$H(e^{j\omega})=\frac{\prod_{j=1}^{M}B_je^{j(\psi_1+\psi_2+\cdots+\psi_M)}}{\prod_{i=1}^{N}A_ie^{j(\theta_1+\theta_2+\cdots+\theta_N)}}=\left|H(e^{j\omega})\right|e^{j\varphi(\omega)}$$

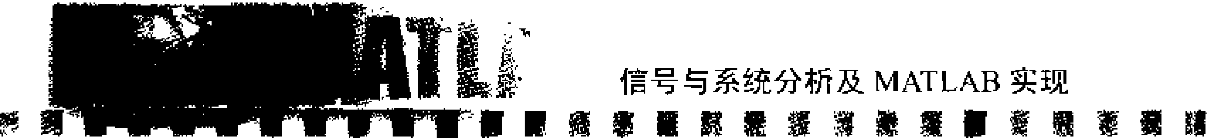
则系统的幅频响应和相频响应为：

$$\left|H(e^{j\omega})\right|=\frac{\prod_{j=1}^{M}B_j}{\prod_{i=1}^{N}A_i} \tag{12-12}$$

$$\varphi(\omega)=\sum_{j=1}^{M}\psi_j-\sum_{i=1}^{N}\theta_i \tag{12-13}$$

由上述分析我们可以得出以下结论：

- 离散系统的幅频响应 $\left|H(e^{j\omega})\right|$，等于系统函数 $H(z)$ 所有零点到单位圆上相角为 ω 的点的距离之积与系统函数所有极点该点的距离之积的比值。
- 离散系统的相频响应等于系统函数所有零点到单位圆上相角为 ω 的点的矢量的相角之和与系统函数所有极点到单位圆上相角为 ω 的点的矢量的相角之和的差



值。

让矢量$e^{j\omega}$沿着单位圆旋转，即数字角频率ω由0～2π进行改变，我们便可直观地求出系统幅频响应和相频响应随的变化，从而分析出系统的频率特性。显然这种变化是以2π为周期的。

根据上述结论，我们可以运用 MATLAB 来实现已知离散系统的零极点分布，求系统频率响应，并绘制其幅频特性和相频特性曲线这一分析过程。

利用 MATLAB 编写实现上述分析过程的通用程序的流程如下：

（1）根据系统函数$H(z)$定义其分子、分母多项式系数向量B和A。

（2）调用前述的 ljdt 函数求出$H(z)$的零点和极点并绘制出系统零极点图。

（3）定义 Z 平面单位圆上的k个频率等分点（对应于单位圆上的k个不同的$e^{j\omega}$点）。

（4）求出$H(z)$的所有零点和极点到这些等分点的距离。

（5）求出$H(z)$的所有零点和极点到这些等分点的矢量的相角。

（6）根据式（12-12）和（12-13）求出单位圆上各频率等分点的$|H(e^{j\omega})|$和$\varphi(\omega)$。

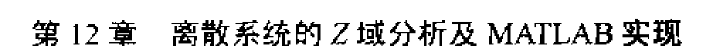
（7）绘制指定范围内系统的幅频特性曲线和相频特性曲线。

下面是实现上述分析过程的 MATLAB 实用函数 dplxy()。该子程序包含四个传入参量，其中k为用户自定义的频率等分点的数目，它的大小决定了 MATLAB 程序绘制的曲线是否能很好地近似实际频率响应曲线，B、A为待分析的系统函数的分子、分母多项式的系数向量，r为程序绘制的频率特性曲线的频率范围，频率范围为 0～r×π，即若r=2，程序则绘制出系统 0～2π 频率范围的特性曲线。

注意

在程序中尽可能地采用了矩阵运算而不是循环运算（例如流程中第（4）、（5）、（6）步的实现），这样可有效提高程序的运算速度。

```
function dplxy(k,r,A,B)
%The function to draw the frequency response of discrete system
p=roots(A);                    %求极点
q=roots(B);                    %求零点
figure(1)
ljdt(A,B)                      %画零极点图
w=0:l*pi/k:r*pi;
y=exp(i*w);                    %定义单位圆上的 k 个频率等分点
N=length(p);                   %求极点个数
M=length(q);                   %求零点个数
yp=ones(N,1)*y;                %定义行数为极点个数的单位圆向量
yq=ones(M,1)*y;                %定义行数为零点个数的单位圆向量
vp=yp-p*ones(1,k+1);           %定义极点到单位圆上各点的向量
vq=yq-q*ones(1,k+1);           %定义零点到单位圆上各点的向量
Ai=abs(vp);                    %求出极点到单位圆上各点的向量的模
Bj=abs(vq);                    %求出零点到单位圆上各点的向量的模
```





```
Ci=angle(vp);                   %求出极点到单位圆上各点的向量的相角
Dj=angle(vq);                   %求出零点到单位圆上各点的向量的相角
fai=sum(Dj,1)-sum(Ci,1);        %求系统相频响应
H=prod(Bj,1)./prod(Ai,1);       %求系统幅频响应
figure(2)
plot(w,H);                      %绘制幅频特性曲线
title('离散系统幅频特性曲线')
xlabel('角频率')
ylabel('幅度')
figure(3)
plot(w,fai)
title('离散系统的相频特性曲线')
xlabel('角频率')
ylabel('相位')
```

以下是用 dplxy()函数对离散系统频率特性进行分析的实例。

例 12-3：已知某离散系统的框图如图 12-12 所示，试用 MATLAB 分析该系统的频率特性，绘制其幅频及相频特性曲线，并分析该系统的作用。

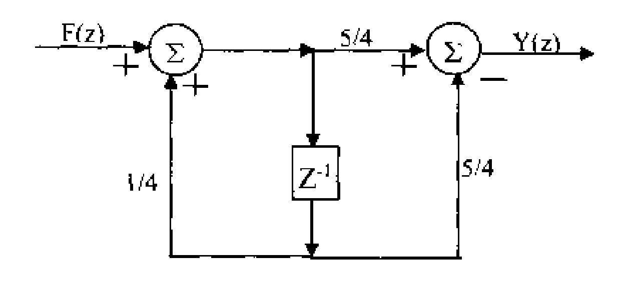


图 12-12　离散系统的框图

这是一个直接型的一阶离散系统方框图，由框图可得系统的系统函数为：

$$H(z)=\frac{5/4(1-z^{-1})}{1-1/4z^{-1}}$$

根据该系统的系统函数，调用前述的求离散系统频率响应的 MATLAB 函数 dplxy，即可绘制出该系统的频率特性曲线。实现该过程的 MATLAB 命令为：

```
A=[1 -1/4];
B=[5/4 -5/4];
dplxy(500,2,A,B)        %绘制系统 2π 频率范围内 500 个频率点的幅频和相频特性曲线及零极点图
```

绘制的系统零极点图、幅频及相频特性曲线分别如图 12-13、图 12-14 及图 12-15 所示。

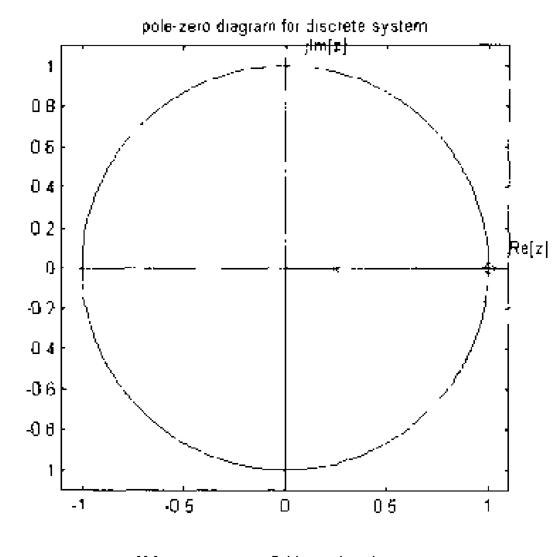


图 12-13　系统零极点图

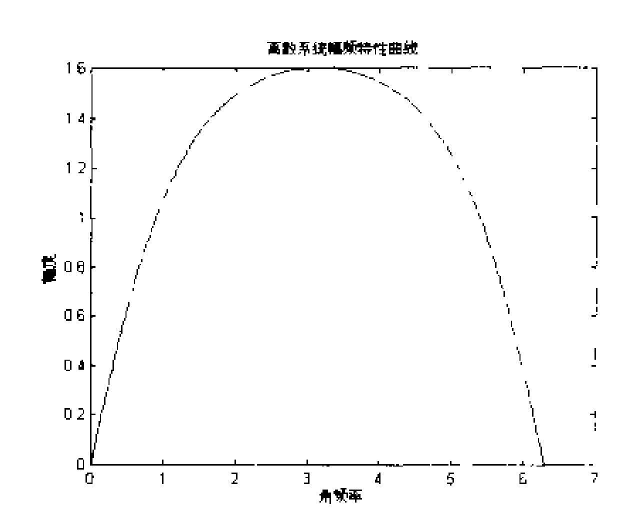


图 12-14　系统幅频特性曲线

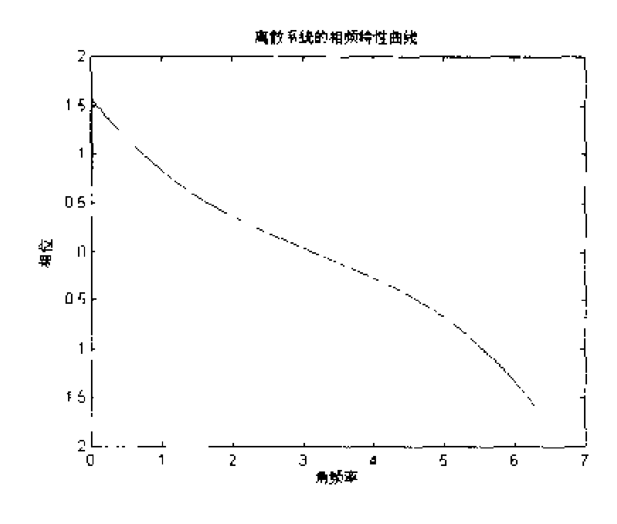


图 12-15　系统相频特性曲线

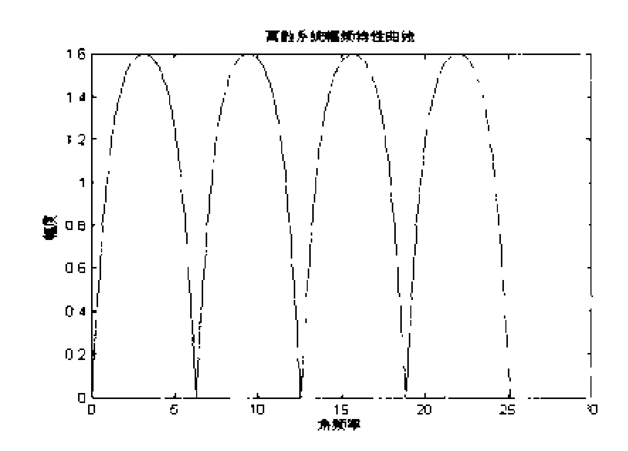


图 12-16　系统幅频特性曲线

从以上几个图中可以看出，该系统呈高通特性，即为一高通滤波器。图 12-16 是该系统 0~8π 范围内的幅频特性曲线，可见系统的频率响应是以 2π 为周期的。

例 12-4：已知某系统的系统函数为 $H(z)=1+5z^{-1}+5z^{-2}+z^{-3}$，试利用 MATLAB 绘出该系统的零极点图及幅频特性曲线，并分析该系统的频率特性。

分析过程：该系统为一全零点型三阶滤波器。调用前述函数 dplxy()即可解决此问题。相应的 MATLAB 命令为：

```
A=[1 0 0 0];
B=[1 5 5 1];
dplxy(800,4,A,B)
```

上述命令绘制的系统零极点图如图 12-17 所示，系统幅频特性曲线如图 12-18 所示。

从绘制的系统幅频特性曲线可以看出，该系统呈低通特性，是一个数字低通滤波器。



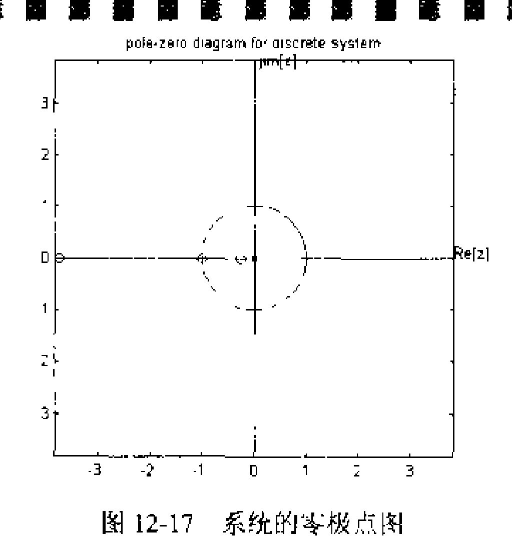


图 12-17　系统的零极点图

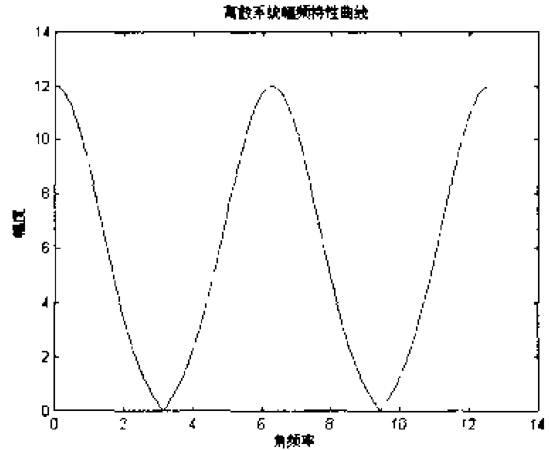


图 12-18　系统的幅频特性曲线

12.5　逆 Z 变换及 MATLAB 实现

离散系列 $f(k)$ 的 Z 变换具有如下一般形式：

$$F(z)=\frac{B(z)}{A(z)}=\frac{\sum_{j=0}^{M}b_j z^j}{\sum_{i=i}^{N}a_i z^i}$$

若 $f(k)$ 为单边序列，即当 $k<0$ 时 $f(k)<0$ 时，则其 Z 变换的收敛域应为 $|z|>\rho_0$，且包括 $z=\infty$，故此时 $F(z)$ 的分子多项式的最高幂次不能高于分母多项式的最高幂次，即满足 $M\leqslant N$。

与拉普拉斯逆变换相类似，逆 Z 变换也可以由部分分式展开法来求得。但要注意的是，离散信号的基本序列是指数序列 $a^k\varepsilon(k)$，其 Z 变换为 $\frac{z}{z-a}$，因此在求逆 Z 变换时，通常并不是直接展开 $F(z)$，而是对 $\frac{F(z)}{z}$ 进行展开。

设某离散序列的 Z 变换为 $F(z)$，则：

$$\frac{F(z)}{z}=\frac{B(z)}{A(z)}=\frac{B(z)}{\prod_{i=1}^{N}(z-p_i)}$$

其中 $p_i(i=1,2\cdots,N)$ 为 $\frac{F(z)}{z}$ 的 N 个极点。若上式满足 $M\leqslant N$，则可对其直接进行部分分式展开得：

$$\frac{F(z)}{z}=\frac{r_1}{z-p_1}+\frac{r_2}{z-p_2}+\cdots+\frac{r_N}{z-p_N}$$

$r_i=(z-p_i)\cdot\left.\frac{F(z)}{z}\right|_{z=p_i}$ （$i=1,2,\cdots N$）称为有理函数$\frac{F(z)}{z}$的留数。

现分两种情况进行讨论。

12.5.1 $F(z)$的所有极点为单实极点

此时，$F(z)=\frac{r_1 z}{z-p_1}+\frac{r_2 z}{z-p_2}+\cdots+\frac{r_N z}{z-p_N}$

则$F(z)$的逆 Z 变换应为：

$$f(k)=\sum_{i=1}^{N} r_i(p_i)^k\varepsilon(k)$$

可见当$F(z)$的所有极点为单实极点时，其对应序列$f(k)$为若干个由$F(z)$极点位置决定的指数序列之和。

12.5.2 $F(z)$有共轭极点

设$F(z)$有一对有共轭极点$p_{1,2}=\alpha e^{\pm j\beta}$，则：

$$\underbrace{F(z)}_{f(k)}=\underbrace{\frac{r_1 z}{z-p_1}+\frac{r_2 z}{z-p_2}}_{f_2(k)}+\underbrace{\frac{r_3}{z-p_3}\quad\cdots\quad+\frac{r_N z}{z-p_N}}_{f_1(k)}$$

其中留数$r_i(i=1,2,\cdots N)$的计算方法与 12.5.1 小节完全相同。即：

$$r_1=(z-p_1)\cdot\left.\frac{F(z)}{z}\right|_{z=p_1}=|r_1|e^{\theta}$$

$$r_2=r_1^*$$

则$f(k)$中由共轭极点所决定的两项复指数序列可以合并为一项，故有：



$$f(k)=f_1(k)+f_2(k)=2|r_1|(\alpha)^k\cos(\beta k+\theta)\varepsilon(k)+\sum_{i=3}^{N}r_i(p_i)^k\varepsilon(k)$$

当 $F(z)$ 具有一对以上共轭极点的情况亦同理。

可见，当 $F(z)$ 有共轭极点时，其对应时间序列将出现按指数规律变化的正弦（或余弦）序列分量。

结论：序列 $f(k)$ 的时域特性完全由其 Z 变换 $F(z)$ 的极点位置决定。

从以上的分析我们可以看出，只要求出 $\dfrac{F(z)}{z}$ 部分分式展开的系数（留数）$r_i(i=1,2\cdots\cdots,N)$，我们就可以直接求出 $F(z)$ 的逆变换 $f(k)$。

与求拉普拉斯逆变换一样，我们也可以用 MATLAB 的 residue()函数来求逆 Z 变换。

设 $$\frac{F(z)}{z}=\frac{B(z)}{A(z)}=\frac{B(z)}{\prod_{i=1}^{N}(z-p_i)}=\sum_{i=1}^{N}\frac{r_i}{z-p_i}+\sum_{j=0}^{M-N}c_jz^j$$

（若 $M\leqslant N$，此项为零）

令 B 和 A 分别是 $F(z)$ 的分子和分母多项式构成的系数向量，则函数：

[r, p, k]=residue (B, A)

将产生三个向量 r、p 和 k，其中 p 为包含 $\dfrac{F(z)}{z}$ 所有极点的列向量，r 为包含 $\dfrac{F(z)}{z}$ 部分分式展开系数 $r_i(i=1,2\cdots,N)$ 的列向量。k 为包含 $\dfrac{F(z)}{z}$ 部分分式展开的多项式项的系数 $c_j(j=1,2\cdots,M-N)$ 的行向量，若 $M\leqslant N$，则 $\boldsymbol{k}$ 为空阵。

用 residue()函数求出 $\dfrac{F(z)}{z}$ 部分分式展开的系数后，便可根据其极点位置分布情况直接求出 $F(z)$ 的逆 Z 变换 $f(k)$。下面将举例说明如何用 MATLAB 来求逆 Z 变换。

例 12-5：已知某离散系统的系统函数为：

$$H(z)=\frac{z^2}{z^2+3z+2}$$

试用 MATLAB 求该系统的单位响应 $h(k)$。

解：

首先利用 MATLAB 对 $\dfrac{H(z)}{z}$ 进行展开，即先考虑：

$$\frac{H(z)}{z}=\frac{z}{z^2+3z+2}$$

然后调用 residue()函数求出 $\dfrac{H(z)}{z}$ 部分分展开的系数和极点，对应的 MATLAB 命令如下：

```
A=[1 3 2];
```

```
B=[1 0];
[r,p,k]=residue(B,A)
```

运行结果为：

```
r =
      2
      -1
p =
     -2
     -1
k =
      []
```

由上述结果可得：

$$H(z)=\frac{2}{z+2}+\frac{-1}{z+1}$$

由于该系统的系统函数只有两个单实极点，故由上述结果可直接求出系统的单位响应为：

$$h(k)=[2(-2)^k-(-1)^k]\varepsilon(k)$$

例 12-6：已知某序列的 Z 变换为：

$$F(z)=\frac{z^2+z}{z^3-2z^2+2z-1}$$

试用 MATLAB 求 $F(z)$ 的逆 Z 变换 $f(k)$。

解：

首选利用 MATLAB 对 $\frac{F(z)}{z}$ 进行部分分式展开，即先考虑：

$$\frac{F(z)}{z}=\frac{z+1}{z^3-2z^2+2z-1}$$

然后调用 residue 函数求出 $\frac{F(z)}{z}$ 部分分展开的系数和极点，对应的 MATLAB 命令如下：

```
A=[1 -2 2 -1];
B=[1 1];
[r,p,k]=residue(B,A)
```

运行结果为：

```
r =
   -1.0000 - 0.0000i
   -1.0000 + 0.0000i
    2.0000
```



```
p =
    0.5000 + 0.8660i
    0.5000 - 0.8660i
    1.0000
k =
      []
```

可见，$\frac{F(z)}{z}$ 包含有一对共轭极点，用 abs() 和 angle()函数即可求出共轭极点的模和相角，命令如下：

```
p1=abs(p')
p1 =
     1.0000      1.0000      1.0000
a1=angle(p')/pi
a1 =
    -0.3333      0.3333           0
```

由上述结果可得共轭极点为：

$$p_{1,2} = e^{\pm j\frac{\pi}{3}}$$

则：

$$F(z) = \frac{-z}{z - e^{-j\frac{\pi}{3}}} + \frac{-z}{z - e^{j\frac{\pi}{3}}} + \frac{2z}{z-1}$$

故：

$$h(k) = [-2\cos(\frac{k\pi}{3}) + 2]\varepsilon(k)$$

上机练习题

1．已知离散系统的系统函数分别如下所示：

（1）　$H(z) = \frac{z^2 - 2z - 1}{2z^3 - 1}$　　　（2）　$H(z) = \frac{z+1}{z^3 - 1}$

（3）　$H(z) = \frac{z^2 + 2}{z^3 + 2z^2 - 4z + 1}$　　　（4）　$H(z) = \frac{z^3}{z^3 + 0.2z^2 + 0.3z + 0.4}$

试用 MATLAB 实现下列分析过程：

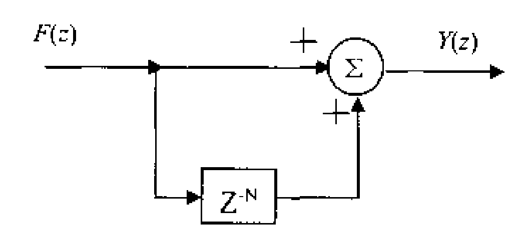
（1）求出系统的零极点位置。

（2）绘出系统的零极点图，根据零极点图判断系统的稳定性。

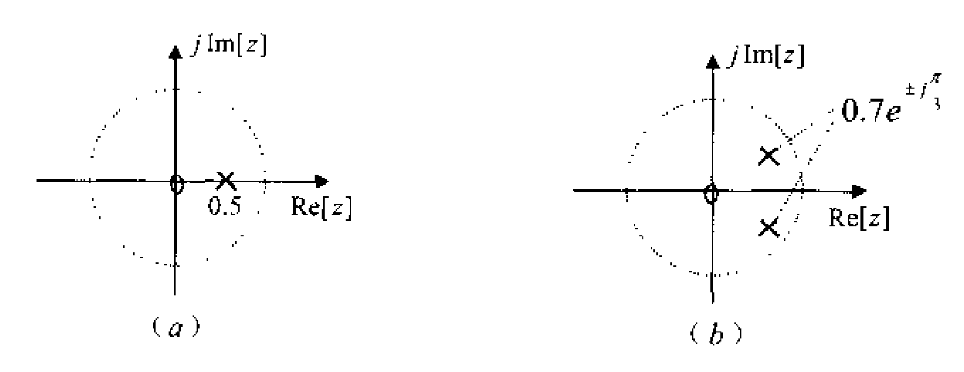
（3）绘出系统单位响应的时域波形，并分析系统稳定性与系统单位响应时域特性的关系。

2．对例 12-2 零极点图所示各系统，试改变（增大或减小）极点模 a 和相角 β 的大小，用 MATLAB 绘制 a 和 β 改变后系统单位响应 $h(k)$ 的时域波形，观察 a 和 β 的大小是如何影响 $h(k)$ 的时域特性的，分析 a 和 β 的值的大小与时域特性的关系。当 $a=1$ 时，$h(k)$ 有何特点？当 $\beta=0$ 和 $\beta \neq 0$ 时，$h(k)$ 时域特性有何不同？

3．已知离散系统的框图如下图所示，试用 MATLAB 绘出系统当 $N=2$、$N=4$ 和 $N=6$ 时的幅频特性和相频特性曲线，并分析系统的频率特性，说明该系统的作用。在该系统中，N 的取值不同将会对系统产生什么影响？



4．已知离散系统系统函数的零极点分布图分别如图（a）和图（b）所示，试根据离散系统频率特性分析的几何矢量分析法的原理，利用 MATLAB 编程绘制出系统的幅频特性和相频特性曲线，并分析系统的作用（虚线部分为单位圆）。



5．已知描述离散系统的差分方程为：

$$y(k)-y(k-1)-y(k-2)=4f(k)-f(k-1)-f(k-2)$$

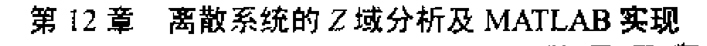
试用 MATLAB 绘出该系统的零极点分布图，并绘出系统的幅频和相频特性曲线，分析该系统的作用。

6．已知因果（单边）离散序列的 Z 变换分别如下所示，试用 MATLAB 求出其逆 Z 变换。

（1）$F(z)=\dfrac{z^2+z+1}{z^2+z-2}$　　　（2）$F(z)=\dfrac{2z^2-z+1}{z^3+z^2+\frac{1}{2}z}$

（3）$F(z)=\dfrac{z^2}{z^2+\sqrt{2}z+1}$　　　（4）$F(z)=\dfrac{z^3+2z^2+z+1}{3z^4+2z^3+3z^2+2z+1}$





7. 已知某离散系统的系统函数如下所示，现令 $\alpha=2$、$\beta=\frac{\pi}{4}$，试用 MATLAB 实现下列分析过程：

$$H(z)=\frac{z^2-2\alpha\cos(\beta)z+\alpha^2}{z^2-2\alpha^{-1}\cos(\beta)z+\alpha^{-2}}$$

（1）绘出系统的零极点分布图，判断系统的稳定性。

（2）绘出系统的幅频及相频特性曲线，并分析系统的频率特性，说明该系统的作用。

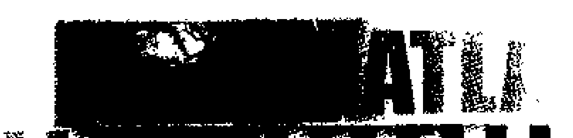
（3）绘出系统的单位响应时域波形。

（4）改变 α 和 β 的取值，重复上述分析过程，分析 α 和 β 的取值对系统频率特性的影响。


读书笔记

附录　MATLAB 常用函数表





MATLAB 常用函数如附表 1 所示。

附表 1 MATLAB 常用函数

函数名	功能	首次出现的章节号
abs	求绝对值或复数求模	2.9
angle	求复数相角	2.9
axis	设置坐标轴范围	5.1.3
cla	清除当前坐标轴	2.5
clc	清除工作空间显示的所有内容	2.5
clear	清除工作空间变量	2.5
clf	清除当前图形	2.5
conv	求离散序列卷积和	2.10
cos	余弦函数	2.9
deconv	解卷积	2.10
diff	符号微分	4.4.2
exp	指数函数	2.9
eye	产生单位矩阵	2.2.2
ezmesh	符号函数三维作图	4.5.2
ezplot	符号函数二维作图	4.5.1
ezplot3	符号函数三维作图	4.5.2
fft	快速傅里叶变换	8.3.3
figure	创建图形窗口	5.3.1
filter	求离散系统响应（差分方程数值解）	7.6
find	寻找矩阵非零元素	6.2.2
fliplr	矩阵反折	2.7
fourier	符号傅里叶变换	9.1
freqs	求连续系统频率响应	10.2
freqz	求离散系统频率响应	12.4
gca	当前坐标轴句柄	5.4.2
gcf	当前图形窗口句柄	5.4.2
get	获取对象属性	5.4.3
global	定义全局变量	3.5
grid	设置网格线	5.1.3
gtext	利用鼠标添加文本标注	5.1.3
Heaviside	产生单位阶跃函数	6.1.1
help	在线帮助	1.4
hold	重叠绘图控制	5.3.2



（续表）

函 数 名	功 能	首次出现的章节号
imag	求复数虚部	2.9
impulse	求连续系统冲激响应	7.3
impz	求离散系统单位响应	7.4
input	键盘输入设置	8.2.2
int	符号积分	4.4.3
length	计算矩阵宽度（列数）	2.7
line	绘直线	5.1.1
linespace	产生线性等分向量	2.3
lsim	连续系统响应仿真	7.5
load	从磁盘装载变量	2.2.3
max	求矩阵元素最大值	2.7
mean	求矩阵元素平均值	2.7
mesh	绘制三维网格曲面	5.2.2
meshgrid	产生栅格数据点	5.2.2
modulate	信号调制	9.3
min	求矩阵元素最小值	2.7
ones	产生值全为1的向量	2.2.2
path	设置 MATLAB 搜索路径	2.5
plot	以折线方式绘制二维图形	5.1.1
plot3	以折线方式绘制三维图形	5.2.1
prod	矩阵求积	2.7
rand	产生随机矩阵	2.2.2
randn	产生正态分布的随机阵	2.2.2
real	求复数实部	2.9
roots	多项式求根	2.10
save	保存变量	2.2.3
sawtooth	产生周期三角波	8.3.2
set	设置对象属性	5.4.2
sign	产生符号函数	6.1.1
simple	符号表达式化简	4.4.4
simplify	符号表达式化简	4.4.4
sin	正弦函数	2.9
size	求矩阵的大小（维数）	2.7
stairs	以阶梯方式绘图	6.1.1
step	求连续系统单位阶跃响应	7.3
stem	绘制离散序列图	5.1.2

（续表）

函 数 名	功 能	首次出现的章节号
subplot	图形窗口分割	5.3.3
subs	符号变量替换	4.3.4
sum	矩阵求和	2.7
surf	绘制三维阴影曲面	5.2.3
sym	定义符号表达式	4.1.2
symadd	符号函数（矩阵）相加	4.3.2
symmul	符号函数（矩阵）相乘	4.3.2
syms	定义符号变量	4.1.2
text	添加文本标注	5.1.3
title	设置图形标题	5.1.3
view	三维图形视角变换	5.2.4
who	查寻工作空间变量	2.5
whos	查寻工作空间变量详细资料	2.5
xlabel	设置横坐标标题	5.1.3
ylabel	设置纵坐标标题	5.1.3
zeros	产生值全为 0 的矩阵	2.2.2

参考书目

1 吴大正，杨林耀，张永瑞．信号与线性系统分析（第三版）．北京：高等教育出版社，1999

2 郑君里，应启珩，杨为理．信号与系统．北京：高等教育出版社，2000

3 吴新余，周井泉，沈元隆．信号与系统——时域、频域分析及 MATLAB 软件的应用．北京：电子工业出版社，1999

4 阎鸿森，王新风，田惠生．信号与线性系统．西安：西安交通大学出版社，1999

5 程卫国，冯峰，姚东等．MATLAB 5.3 应用指南．北京：人民邮电出版社，1999

6 楼顺天，李博菡．基于 MATLAB 的系统分析与设计——信号处理．西安：西安电子科技大学出版社，1999

7 陈怀琛，王朝英，高西全等译．数字信号处理及其 MATLAB 实现．北京：电子工业出版社，1998

8 刘树棠译．信号与系统计算机练习——利用 MATLAB．西安：西安交通大学出版社，1999

9 高俊斌．MATLAB 语言与程序设计．武汉：华中理工大学出版社，1998

10 张志涌等．精通 MATLAB 5.3 版本．北京：北京航空航天大学出版，2000